NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

APPLIED CYBER OPERATIONS CAPSTONE PROJECT REPORT

TEST AND EVALUATION OF THE MALICIOUS ACTIVITY SIMULATION TOOL (MAST) IN A LOCAL AREA NETWORK (LAN) RUNNING THE COMMON PC OPERATING SYSTEM ENVIRONMENT (COMPOSE)

by

Aaron M. Littlejohn
Ehab Makhlouf

September 2013

Thesis Advisor: Gurminder Singh
Second Reader: Arijit Das

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704–0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202–4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704–0188) Washington DC 20503.			
1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** September 2013	**3. REPORT TYPE AND DATES COVERED** Capstone Project Report	
4. TITLE AND SUBTITLE TEST AND EVALUATION OF THE MALICIOUS ACTIVITY SIMULATION TOOL (MAST) IN A LOCAL AREA NETWORK (LAN) RUNNING THE COMMON PC OPERATING SYSTEM ENVIRONMENT (COMPOSE)		**5. FUNDING NUMBERS**	
6. AUTHOR(S) Aaron M. Littlejohn, Ehab Makhlouf			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943–5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. government. IRB protocol number ____N/A____.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited		**12b. DISTRIBUTION CODE** A	

13. ABSTRACT (maximum 200 words)

In the Department of the Navy's layered defense approach to protecting a computer network, it is the network's administrators who provide the last layer of defense before the end user. Training network administrators is a rather expensive and time consuming process. However, this training needs to be provided on a regular basis in order to refresh their readiness and to train them to respond to new, emerging threats. Malicious Activity Simulation Tool (MAST) aims to provide realistic, tailored simulation of malicious activity for the purpose of training network administrators to recognize and respond to threats on the network they manage.

In a continuation of MAST development, this thesis reports the testing and evaluation of the MAST functionality on a Local Area Network (LAN) using a Common PC Operating System Environment (COMPOSE) as its network operating system. We conclude that MAST can present realistic simulations of malicious activity that could be detected, recognized, and responded to by network administrators and host network, while posing no threat to the operational readiness of the host network or its supported missions.

14. SUBJECT TERMS Defense, Simulation, Network Administrator Training, Cyberspace, Cyber Domain, Cyber Test Range			**15. NUMBER OF PAGES** 107
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540–01–280–5500 Standard Form 298 (Rev. 2–89)
Prescribed by ANSI Std. 239–18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

TEST AND EVALUATION OF THE MALICIOUS ACTIVITY SIMULATION TOOL (MAST) IN A LOCAL AREA NETWORK (LAN) RUNNING THE COMMON PC OPERATING SYSTEM ENVIRONMENT (COMPOSE)

Aaron M. Littlejohn
Lieutenant Commander, United States Navy

Ehab Makhlouf
Lieutenant, United States Navy

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN APPLIED CYBER OPERATIONS

from the

NAVAL POSTGRADUATE SCHOOL
September 2013

Reviewed By: Gurminder Singh
Capstone Project Advisor

Arijit Das
Capstone Project Co-Advisor

Approved By: Cynthia Irvine,
Chair, Cyber Academic Group

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

In the Department of the Navy's layered defense approach to protecting a computer network, it is the network's administrators who provide the last layer of defense before the end user. Training network administrators is a rather expensive and time consuming process. However, this training needs to be provided on a regular basis in order to refresh their readiness and to train them to respond to new, emerging threats. Malicious Activity Simulation Tool (MAST) aims to provide realistic, tailored simulation of malicious activity for the purpose of training network administrators to recognize and respond to threats on the network they manage.

In a continuation of MAST development, this thesis reports the testing and evaluation of the MAST functionality on a Local Area Network (LAN) using a Common PC Operating System Environment (COMPOSE) as its network operating system. We conclude that MAST can present realistic simulations of malicious activity that could be detected, recognized, and responded to by network administrators and host network, while posing no threat to the operational readiness of the host network or its supported missions.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ACK	Acknowledge
APT	Advanced Persistent Threat
C2	Command and Control
CND-OSE	Computer Network Defense-Operating System Environment
COMPOSE	Common PC Operating System Environment
CONOPS	Concept of Operations
CPU	Central Processing Unit
DC	Domain Controllers
DCO	Defensive Cyber Operations
DDoS	Distributed Denial of Service
DGO	Department of Defense Global Information Grid Operations
DHCP	Dynamic Host Configuration Protocol
DISA	Defense Information Systems Agency
DNS	Domain Name Service
DoD	Department of Defense
DoS	Denial of Service
DON	Department of the Navy
EICAR	European Institute for Computer Anti-Virus Research
ePO	e-Policy Orchestrator
FIN	Finish
FISMA	Federal Information Security Management Act
FTP	File Transfer Protocol
GIG	Global Information Grid
GUI	Graphical User Interface
HBSS	Host-Based Security System
HIPAA	Health Information Portability and Accountability Act
HIPS	Host-based Intrusion Prevention System
HTTP	Hypertext Terminal Protocol
IASE	Information Assurance Support Environment
IC3	Internet Crime Complaint Center

IRC	Internet Relay Chat
IP	Internet Protocol
IPS	Intrusion Prevention System
IT	Information Systems Technician
ITACS	Information Technology and Communications Services
LAN	Local Area Network
LOO	Lines of Operation
MAC	Media Access Control
MAST	Malicious Activity Simulation Tool
MMORPG	Massively Multiplayer Online Role-Playing Game
MMS	Mission Management Server
MOA	Memorandum of Agreement
MS	Microsoft®
MSSQL	Microsoft® SQL Server®
NCOR	Navy Cyber Operations Range
NIOC	Navy Information Operations Command
NMAP	Network Mapping Protocol
NPS	Naval Postgraduate School
OCO	Offensive Cyber Operations
OPNAV	Office of the Chief of Naval Operations
OSI	Open Systems Interconnection
PII	Personally Identifiable Information
PMW-130	Information Assurance and Cyber Security Program Office, SPAWAR
PMW-160	Tactical Networks Program Office, SPAWAR
PSH	Push
QDR	Quadrennial Defense Review
RDML	Rear Admiral, Upper Half
RFC	Request for Comment
RST	Reset
SCADA	Supervisory Control and Data Acquisition
SCCVI	Secure Configuration Compliance Verification Initiative

SCRI	Secure Configuration Remediation Initiative
SE	Simulation Execution
SECDEF	Secretary of Defense
SG	Simulation Generation
SPAWAR	Space and Naval Warfare Systems Command
SSC	SPAWAR Systems Center
SYN	Synchronize
TDSC	Training Development and Support Center
TCP	Transmission Control Protocol
TTP	Tactics, Techniques, and Procedures
URG	Urgent
URL	Uniform Resource Locator
USCYBERCOM	U.S. Cyber Command
VTC	Video Teleconference
VTE	Virtual Training Environment
WINS	Windows Internet Naming Service

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

We would like to express sincere appreciation to Dr. Gurminder Singh, Mr. Arijit Das, and Mr. John Gibson for their mentorship and patience during our research. Their direction, guidance, and expertise were essential to the success of this project. These gentlemen invigorated our drive during the challenging portions of our research with their genuine concern for advancing and improving upon the capabilities and technology available to the war fighter.

In addition, we wish to acknowledge, and are grateful for, the contribution of other key contributors to our work. Dr. Rachel Goshorn, as the Naval Postgraduate School's Liaison to the Space and Naval Warfare Systems Command (SPAWAR), cleared obstacles and helped us establish contacts within their different program offices. From the Tactical Networks Program Office (PMW-160), we thank Commander Gerald Most for his generous support of our research through the provision of the Common PC Operating System Environment (COMPOSE) installation package and distant support from his COMPOSE installation engineer. From the Information Assurance and Cyber Security Program Office (PMW-130), we thank Ms. Susan Hood, Mr. Shadi Azoum, and Ms. Bingh Doung for their generous support of our research through the provision of the Computer Network Defense – Operating System Environment (CND-OSE) installation package, Host Based Security System (HBSS) manuals, and distant support. Both PMW-130 and PMW-160 provided much needed assistance in acquiring, installing, and configuring software without which our research would have never occurred. In addition, we thank PMW-130 for their patience with our many inquiries about the HBSS functionality during the development of the Simware modules. To the fine officers and sailors of the Navy Information Operations Command (NIOC) in Norfolk, Virginia, with whom we worked with during our research of and testing on the Navy Cyber Operations Range (NCOR), we thank you for assistance and patience with us. Specifically, Lieutenant Steven Calhoun and Mr. James Powell were vital to our successful testing on the NCOR.

This work would not be possible without the great contributions of the MAST family of researchers: William Taff, Paul Salveski, Justin Neff, Jim Hammond, Ray Longoria, Nathan Hayes, Greg Belli, and Erik Lowney. Finally, we would like to thank our community mentor, Captain Danelle Barret, whose guidance, integrity, leadership, and example have made us better leaders.

To all contributors mentioned, and not, we thank you for your selfless support, your professionalism, and your service to our great country.

From Ehab: I would like to thank my thesis partner, Aaron Littlejohn, whose support, dedication, and friendship made this project a success. Last, I would like to thank my wife, Victoria, and my son, Ahab, who have walked this path with me and have sacrificed much.

From Aaron: I thank Ehab for every time he smiled and agreed with me even when he felt I was wrong. His technical expertise and leadership skills are much needed in the execution of our nation's missions in cyberspace. I look forward the next opportunity to serve with him. I thank my wife, Kristy, for her devotion to me, our family, and our country. To the three little gentlemen who always complain when I have to leave but always greet me with a big smile when I come home, the world is a better place because you are in it.

I. INTRODUCTION

Information security proficiency among many Department of Defense (DoD) network administrators, both afloat and ashore, is lacking [1]. In the face of the growing threats to our information systems, which become more and more essential to mission execution throughout the DoD, our network administrators must be able to recognize and respond to malicious activity. Due to budget constraints, network administrators are provided initial network administration training with cursory network security training. Beyond the this cursory network security training, a small percentage of these network administrators are given intermediate training on network security as it pertains to network defense (e.g., configuration of their network's Access Control List, local security policies, and global policy objects). Once trained in network administration and the establishment of network security through defense-in-depth measures, members of the U.S. Navy's Information Systems Technician (IT) work force is expected to continue developing their skills and improving upon skill sets established while attending Navy-run IT schools. There is no provision for training network administrators to recognize and properly respond to malicious activity on the networks they administer. The Malicious Activity Simulation Tool (MAST) was designed to remedy this shortfall in training.

MAST was incrementally designed and developed at the Naval Postgraduate School (Taff and Salveski [2], Neff [3], Longoria [4], Hammond [5], Hayes [6], Belli [7], and Lowney [8]) to provide commanding officers, network security instructors and network security evaluators with the ability to deliver a simulation of malicious activity to their target audience on the very network they are supposed to manage. Whether for initial development of network security skills, continued training in network security or evaluation of network security skills in the IT work force, MAST aims to provide a realistic, tailored simulation of malicious activity. MAST is designed to be flexible enough to meet the intent of the training as well as the ability of the training audience. Because network administrators must be ready to respond to malicious activity on their networks, MAST should be able to run its simulations on the same network without

causing a reduction in the network's operational readiness or availability. We discuss this further in Chapter III.

A. OBJECTIVES

The purpose of this research is to test and evaluate the functionality of MAST in execution of its simulations on a LAN utilizing the COMPOSE variant as its operating system. Our tests and evaluations determined MAST is capable of executing the defined simulations and is compatible with the COMPOSE operating system. Common PC Operating System Environment (COMPOSE) is a bundle of software, managed by The Space and Naval Warfare Systems Command (SPAWAR). COMPOSE software suite includes: the operating system, and all software applications. COMPOSE provides afloat platforms with a consolidated software suite that meets the Navy's operational requirements.

Guided by the test plan presented in [5], there are two overall objectives for our quantitative testing of MAST:

- Assess the compatibility of MAST with the COMPOSE environment, both when idle and while executing Simware modules
- Verify the functionality of the MAST Kill Switch on a physical network

All functions, which MAST executes in the delivery of its simulations, were evaluated. Our research created a two-fold process of testing and evaluating not only the functions of MAST as they pertain to its loading, configuration capabilities, execution of malware simulations, hereinafter referred to as Simware, and termination of said Simware, but also as it pertains to the compatibility of these functions with the rest of the applications within a COMPOSE operating system and the operating system itself.

Our research focused solely on the whether or not MAST met its functional requirements on the Department of the Navy's (DON) COMPOSE operating system and none others. It is not within the scope of this thesis to begin any certification and accreditation efforts. Confidence in the MAST program's ability to function on a LAN using the COMPOSE operating system was established through a comparison of the

functional requirements that shaped the MAST program's design to the functionality displayed by the MAST program in a test environment.

B. METHODOLOGY

Our test methodology was influenced by the Operational Test Director's Manual (COMOPTEVFORINST 3980.2) [9] and drew from the quantitative testing process presented by Lieutenant Hayes in his thesis [6]. We executed our tests of MAST, its Simware modules and its Kill Switch, in two separate environments. First, we built a test and evaluation network at Naval Postgraduate School that was configured as a U.S. Navy shipboard network using COMPOSE as the operating system. By complying with DoD and DON configuration guidance, this test and evaluation network, referred to henceforth as the MAST Test Bed, replicated an operational Navy shipboard network. Second, we validated our test findings by executing the same tests of MAST in a DoD-certified and accredited environment, the Navy Cyber Operations Range (NCOR).

This thesis conducted an evaluation of network behavior during malware injection simulation. The thesis verified COMPOSE compatibility by:

- Inspecting application, security, and system logs on each server and workstation while MAST is idle and while MAST is executing Simware modules.
- Verifying the functionality of the MAST Kill Switch.
- Verifying the proper de-installation and removal of MAST.

The thesis researched and documented host and network resource usage to include memory and processes resources. Data was collected from each workstation and server in accordance with the test methodology suggested in [6]. Further explanation of methodology, data collection, and analysis is provided in Chapter IV.

C. BENEFITS OF THIS RESEARCH TO THE DOD/DON

In the DON's layered defense approach to protecting a computer network, it is the network's administrators who manage the last layer of defense before the end user. Training network administrators is a rather expensive and time consuming process but this training needs to be provided on a regular basis to refresh their readiness and to train

them to respond to new, emerging threats. MAST aims to provide a realistic, tailored simulation of malicious activity for the purpose of training network administrators to recognize and respond to threats on the networks they manage. It can also be used as a network security evaluation tool generating scenarios similar to ones executed by DoD Red Teams. This research established the value of MAST in the context of how it may aid the DON and DoD efforts to improve the training of its network administrators in a time of great need for more capable network defenders able to ensure the freedom of maneuver in cyberspace coupled with a time of fiscal judiciousness and an austere funding environment throughout the DoD. The benefits of this research to the DoD and DON are further explained in Chapter II.

D. ORGANIZATION

The remainder of this document proceeds as follows:

Chapter II establishes a foundation for this research by providing information about the potential threats to not just DoD networks, but all networks connected to the Internet. It discusses the fundamental network security objectives of confidentiality, integrity, and availability as well as how malicious actors or threat agents maliciously compromise these objectives to gain access to a network, its users, and the information contained therein. It also historically explains how MAST has progressed toward a solution that will enable the DoD to better equip network administrators to identify and respond to attacks from these threat agents.

Chapter III describes the design considerations of our experimentation. We take a deeper look at the elements required to produce a realistic, tailored simulation of malicious activity with the intent of facilitating effective training for operational readiness in cyberspace. In order to better understand the motivation behind the design of MAST, we explore the shortfall of network self-defense that exists in the DoD. As a castle requires both layers of defense as well as the ability to dynamically defend itself against whatever manner of attack threatens the security and safety of its occupants, so too does a DoD network in order to protect its users and information. Today's networks possess defense-in-depth architectures governed by policy, regulated by appointed

institutions, and routinely assessed through inspections and evaluative exercises. What is missing is the ability to dynamically defend the network. DoD network administrators require a training environment where they can encounter inert examples of known malicious behavior against which they can develop and strengthen their cyber self-defense tradecraft. Chapter III explains how MAST aims to deliver this environment. It also describes the design of the testing environments used to evaluate MAST's compatibility with the COMPOSE environment.

Chapter IV discusses in detail the methodology of our test and evaluation of the functionality of MAST system on our two test environments. In it, we explain the efforts made to ensure the MAST Test Bed replicated an operational shipboard network. In addition, we describe our investigative efforts to ensure the realistic simulation of malicious activity drove the development of both Simware modules. We also capture the logical and physical differences between our two testing environments: The MAST Test Bed at the Naval Postgraduate School and the Navy Cyber Operations Range (NCOR) at the Navy Information Operations Command (NIOC) in Norfolk, VA. Finally, we present the analysis and results from our tests in both environments.

Chapter V summarizes the discussions, descriptions and explanations made throughout this paper. It provides the conclusion drawn from the results of our test and evaluation of MAST. It provides a summary of recommended measures that should be addressed or resolved before a commander would be inclined to allow the MAST program to be installed on their operational network. It concludes with several recommendations for future work to include continued collaboration with the NCOR and further advancements in malicious activity simulation.

THIS PAGE INTENTIONALLY LEFT BLANK

II. BACKGROUND

Disreputable activities on the Internet vary in complexity from simple and primitive to advanced, persistent, and calculated. In order to better understand how the Simware used in our research creates a mock representation of the threats faced by our networks, we must properly explain the disreputable side of the Internet. To do that, we begin by identifying and defining the three fundamental objectives of network security. Network security personnel focus on these objectives when shaping their plans for defending their network. Nefarious individuals focus on these objectives for polar opposite reasons. As an example, a network administrator must take measures to ensure the confidentiality of sensitive information stored on his network. An attacker wanting to gain access to this sensitive information seeks to violate that confidentiality.

With an understanding of the core network security objectives, we explain the concept of a threat. We describe the type of person who acts on these threats. We then provide an overview of the types of malicious software, or malware, and malicious activity employed by these attackers. These explanations give a context to the simulated malicious environments we create with MAST's Simware. We close the chapter with a brief discussion of the works of all previous MAST project research phases.

A. THE FUNDAMENTAL OBJECTIVES OF NETWORK SECURITY

Whether discussing physical security or network security, all security programs are shaped around a balance between the organization's efforts to execute its missions and to provide assurance in the tools needed to conduct mission-supporting operations. A military command is similar to an investment bank or any other civilian business in that it seeks a balance between securing the information and information systems required to execute its assigned missions and not hindering its ability to execute assigned tasking due to implementation of excessive security measures. Some organizations may have specific, legally mandated, security requirements. For example, health care providers must implement security measures in accordance with the Health Information Portability and Accountability Act (HIPAA) of 1996 or face stiff penalties. The end decision on how an

organization will balance its business goals, security goals, and security requirements shapes the objectives of its security policies. These security objectives vary from one organization to the next in size and complexity. All objectives, regardless of the organization, address one or more of the fundamental objectives of security: confidentiality, integrity, and availability. All three objectives are defined in Title III of the E-Government Act, also known as the Federal Information Security Management Act (FISMA) [10].

1. Confidentiality

Ensuring confidentiality of data means preventing the unauthorized access to or disclosure of sensitive information, whether personal or proprietary in nature [10]. Security measures that address confidentiality protect sensitive data at rest as well as while in transit. They also address protection of the systems that process and contain the data and even the facilities containing these systems.

2. Integrity

When you provide for the integrity of information you guard against the unauthorized modification or destruction of information [10]. While a confidential network ensures sensitive data is created, transferred, and stored in a manner that prevents unauthorized access or disclosure, a network ensures that data's integrity by guarding against any unauthorized manipulation, whether intentional or unintentional.

3. Availability

Ensuring the availability of information requires provisions for access to that information in a timely and reliable manner [10]. Unauthorized disruptions in access to information or information systems is a violation of the availability principle.

Network administrators will take actions to ensure the confidentiality, integrity, and availability of the information and information systems relied upon by their organizations. They will deploy firewalls at their network perimeters to prevent

unauthorized access to their network. They will require users to encrypt e-mail with the recipient's public key when sending attachments to protect the confidentiality of the attachment in transit. They will ensure their anti-virus signature definitions are up to date in order to detect malware designed to attack the availability of resources like computer processing and network bandwidth. These measures are pro-active in nature and are intended to protect the confidentiality, integrity, and availability of the organization's information and information systems from the attackers who would exploit them.

B. THREATS AND THREAT AGENTS

As an organization shapes its security objectives to provide for confidentiality, integrity, and availability of its assets, it must take into consideration several factors. The security team must identify:

- Any vulnerabilities in their assets
- Any threats that can exploit these vulnerabilities
- The risk posed by these vulnerabilities to the organization
- The countermeasures the organization will employ to offset these vulnerabilities

Shon Harris's book, *CISSP All-in-One Exam Guide*, is a good source of fundamental security definitions [11]. As such, our following definitions are derived from hers. Vulnerabilities are any weakness that may enable an attacker to violate the confidentiality, integrity, or availability of information or information systems. Vulnerabilities may be present in the way information is created, processed, transmitted, stored, or destroyed. They may be present at any layer of the Open Systems Interconnection (OSI) Model. A threat is the potential danger posed to the information or information systems by a vulnerability. A threat may be internal or external to the information system or network. Threats may be executed by either benign or malicious agents. A system administrator who does not know how to properly configure the system security mechanisms is a benign threat, whereas an attacker is malicious. There must be intent for someone to be considered an attacker. The risk associated with a vulnerability is the likelihood a threat agent will act on the vulnerability. Risk also takes into consideration the business impact this action would have on the organization. Anything

put in place to mitigate risk posed by a vulnerability is called a countermeasure. A countermeasure may eliminate the vulnerability or reduce how affective a threat agent would be at exploiting the vulnerability.

To explain these terms in context, consider the vulnerability of a laptop not using the latest virus signatures database for its anti-virus software. The vulnerability is the system is susceptible to infection. The threat is the potential harm the laptop or information stored on it faces by an undetected virus. The threat agent is the mechanism by which the virus is introduced into the system. The risk takes into consideration what damage the malicious code could do to the laptop's confidentiality, integrity, and availability as well as what that means to the organization. The business impact portion of risk plays an important role in the determining which countermeasures to employ.

In the last decade, the information security environment has changed. Information security providers have reported a marked increase in the number of targeted attacks on information systems and networks. Attacks may come from epidemic, resource-impeding malware such as the Conficker worm [12] or may be directed at specific targets with specific intent. These attacks come from threat agents or attackers that may be grouped into several different categories. During his keynote presentation to the 2012 Hack in Paris conference, Mikko Hypponen, Chief Research Officer for the Internet security company F-Secure, explained that most cyber security attacks originate from three sources: criminals, hacktivists, and governments [13].

1. Criminals

The Internet continues to become more and more a part of normal, everyday life for most citizens of the world. Parents purchase school clothes on-line for their children. Students register for classes. Friends stay connected across great geographical distances via social media, video-enabled web conferencing, and massively multiplayer online role-playing games (MMORPG). During all of these Internet-based exchanges and transactions, the Internet user must establish his identity by providing credentials. Once the user's identity is authenticated, it is assumed any transactions conducted are made by the actual user. The credentials establish the integrity, and in some cases the

confidentiality, of these transactions. If an unscrupulous individual or organization came into possession of the user's credentials, they could conduct fund transfers from the user's bank accounts, access personally identifiable information (PII), open additional lines of credit in the user's name, and even purchase goods with the fraudulently established new line of credit. Any Internet-based transaction that requires a person to authenticate himself could be conducted under the ruse that the attacker was the victim. This is what cyber criminals do. They exploit our reliance on the Internet for their own ill-gotten gains. Cyber criminals vary in skill level and citizenship. The line distinguishing cyber criminals from government and hacktivists is the motivation behind their malfeasance. In its 2011 Internet Crime Report, the Internet Crime Complaint Center (IC3) stated there were more than 300,000 complaints of online criminal activity in the United States for that year [14]. In addition to conducting malicious activity for the purpose of making profit, some cyber criminals and criminal organizations make themselves available for hire to anyone who wishes to use their nefarious tradecraft. Not limiting their skill set to just identity theft, cyber criminals may be hired to conduct industrial espionage, denial of service (DoS) attacks, and fraudulent manipulation of data stored in computer-based records.

2. Hacktivists

The Vietnam War produced a well-documented time of political activism in U.S. history. Political activists expressed disapproval of the War and questioned its legality over a wide spectrum of activity. Protest events ranged from peaceful sit-ins and marches to displays of violence. More recently in history, the infusion of computing technology and society has enabled another venue for expression of support of, or opposition to, political issues. Impedance of the confidentiality, integrity, or availability of the Internet or Internet-based services in support of political activism is known as hacktivism. François Paget, a senior malware research engineer at McAfee Labs, wrote a white paper about hacktivism in 2012. In it, Paget identifies key dates in the origin of hacktivism to include [15]:

- The 1981 establishment of The Chaos Computer Club in Berlin
- The WANK (Worms Against Nuclear Killers) worm that attacked a NASA network in 1989
- The hacker group UrBan Ka0S attacks of Indonesian government websites to protest oppressive conditions in Timor
- The 2001 DoS attack on the Lufthansa website in protest of the German government's use of its aircraft to deport undocumented migrants out of Germany.

Paget identifies three major groups of hacktivists and provides examples of each. The first group is a specific group, Anonymous. Anonymous began as a group of Internet Relay Chat (IRC) personas who put their hacking skills to use as they protested perceived overstepping of moral boundaries. Their use of malicious activity to present their political case mostly consisted of degrading or denying access to websites through DDoS attacks [15]. A targeted website would receive a large quantity of queries from workstations under the control of Anonymous members. The workstations launched these queries using an open-source, DoS creation tool known as the Low Orbit Ion Canon (LOIC) [15]. The volume of queries would saturate the website's infrastructure making it unavailable to legitimate web traffic queries. Their numbers grew as the Anonymous movement shifted from politically-motivated pranks to main-stream hacktivism.

The hacktivists of the Arab Spring embody Paget's second hacktivism group, the cyberoccupiers. Cyberoccupiers rely on Internet-based social networks to present their political views, spread propaganda, and recruit additional members [15]. Unlike Anonymous, cyberoccupiers do not use denial or degradation of services as a means to present their case. In fact, they rely on such services to conduct their form of hactivism. Cyberoccupiers used social media sites as the planning vehicle for protests in countries like Tunisia, Egypt, and Libya. When the governments of these countries attempted to control the world media's coverage of these events, cyberoccupiers in these countries and around the world collaborated to get audio and video files on the Internet, giving a globalized voice to their cause. Whereas cyberoccupiers are motivated by their loss of support for a ruling political power, the third hacktivism group contrasts their motivation.

Paget's third hacktivism group is called the patriots and cyber warriors. In calling this third group "cyber warriors," Paget is not implying an official, state-sponsored military organization. Instead, these hacktivists behave like fundamentalists These hacktivists thrive in totalitarian countries and conduct attacks as expression of support for national and extremist moments [15]. The 2007 cyber-attacks on Estonia and the 2008 cyber-attacks on Georgia were reportedly perpetrated by patriotic Russian hackers. In the case of Estonia's cyber-attack, the Distributed Denial of Service (DDoS) attacks on Estonia's banking, government, and media websites were coordinated and executed by the Nashi, a Russian youth organization with political ties to the Putin administration.

3. Governments

Hacktivists and, at times, criminals are willing to openly acknowledge their conduct of cyber-attacks. In contrast, there are few, if any, government organizations willing to publicly acknowledge their own conduct of cyber-attacks. Although proof of government sponsorship of malicious activities executed in support of national security policies may exist at a higher classification, this is an unclassified thesis and, therefore, will not address government cyber intelligence or its collection. At the unclassified level, open media provides that the identification of governments as the third source of most cyber-attacks stems from intelligence collected by private industry. Network security solution providers like F-Secure based out of Finland, McAfee and Symantec based out of the U.S., and Kaspersky Labs headquartered in Russia, are in the business of identifying malware present on the Internet, developing anti-virus signatures to facilitate detection and prevention of malware, and learning how to remove the malware from the infected machines of their clients. The origin of malicious code, and with some inference, the creator of the malicious code, may be revealed during the analysis process. This is how cyber experts like F-Secure's Mikko Hypponen can implicate governments as being a source of cyber-attacks when no government has formally acknowledged such actions. One such formal implication was made by Mandiant, a cyber security firm, in February, 2013. In their report, they identify an advanced persistent threat (APT) responsible for computer security breaches across the globe. The report contains forensic evidence that the Chinese government may be directly linked to this threat [16].

C. MALICIOUS SOFTWARE

Computing devices utilize code to provide the user with capabilities. Installed software applications (e.g., web browsers, on-line video chat tools, word processing programs), internal and external hardware (e.g., memory chips, internal wireless network interface cards, video cards), and attached peripheral devices (e.g., keyboard, monitor, mouse, printer) all require the presence or absence of an electric signal, which translates to code that tells them what to do. Firmware and software programs provide the computer's processor with input data. The processor is the brain of all computing functions, whether in a personal computer or in the Supervisory Control and Data Acquisition (SCADA) equipment used to control flow valves at municipal water treatment plants. The functional relationship between the processor brain and the appendages of software and hardware relies on an assumption that all signals sent and received are not manipulated or interrupted in any manner. In other words, the signal integrity is intact. If a program can be written to accomplish productive work, a program can also be written to accomplish destructive work.

It is possible to create software that produces a desired effect, malicious in nature, by exploiting the electric signals tied to computing technology. In 2005, Troy Nash, a member of the Vulnerability and Risk Assessment Program at Lawrence Livermore National Laboratory, published a case study on ways to improve control system security. In this report, he provides his definition of malicious software, or malware, as:

> Programming (code, scripts, active content, and other software) designed to disrupt or deny operation, gather information that leads to loss of privacy or exploitation, gain unauthorized access to system resources, and other abusive behavior. [17]

One example of malware is a computer virus. To detect and protect a computer against infection from a computer virus, network administrators employ anti-virus software. Anti-virus software relies on virus signatures to identify and quarantine or remove malware before the file is allowed to access or remain on the protected network, work station, or device. A virus signature is a copy of the malicious code or DNA of a virus [18]. A virus, or any other malware, for that matter, is detected by two means, generic

detection and heuristic detection. In generic detection, anti-virus software scan a file and compare it with existing signatures of known viruses. In heuristic detection, the virus is not detected by its signature but by its behavior when executing its malicious intent. Unfortunately, modern malware designers are becoming more adroit at obfuscating of their malicious code. Code obfuscation involves transforming the program to hide its intent while maintaining its intended function [18]. Obfuscated malware, Virus A, for example, avoids detection by anti-virus software because it looks different than the virus signature for Virus A loaded into the anti-virus software's signature database. Malware may also go undetected by anti-virus software scans if the virus signature for that virus has not be created. In order for a signature to be created, an anti-virus company must know of the exploit and write a signature based on its characteristics.

Whether undetected because of obfuscation or the lack of virus signature, a virus may still be identified by its functionality. A network administrator does not necessarily need a report from the network's anti-virus program to know a worm may be the cause behind a significant degradation in available network bandwidth. Malware may be grouped into several categories. Provided below are the general functionalities of each. Identifying the presence of malware on a workstation based on its functionality is not an easy feat. Like a human ailment, malware "symptoms" or functionality may span several of these categories making it difficult to identify and treat [19].

1. Virus

A virus is self-replicating program that attaches to files in the infected machine, often for the purpose of corrupting or deleting files on that machine. It replicates in order to facilitate spreading from one file to another on the same machine. The spread of a virus from one computer to another requires the transfer of the infected file via contact through removable storage media, network connectivity, or the Internet.

2. Worm

Like a virus, a worm is self-replicating. Unlike a virus, it does not depend on a host file to function. Its purpose is propagation not destruction. As mentioned earlier, while the worm replicates and spreads from machine to machine, it may be detected by

the increasing amount of network bandwidth absorbed for the act of propagation. The more are machines infected by the worm, the more worm files are generated. The more worm files generated, the more files propagated to other network devices. Robert Morris, a first-year computer science graduate student at Cornell University, infected several thousand computers throughout the United States with a worm he created to exploit security flaws in the UNIX operating system. [20].

3. Downloader

The main purpose of a downloader file is to download and install files. These files may be additional malware used by an attacker to further establish unfettered access to the infected workstation and, as a result, the associated network. In order for this malware type to be successful, the infected workstation must have connectivity to the site from which the downloader will download additional malware [19].

4. Launcher

A launcher program activates other malware. An example of this is a Trojan horse, a program, usually legitimate, that is modified to contain malicious functionality. The Trojan horse, when activated, launches the payload malware contained within. The payload itself is usually also malicious in nature. As a result, network administrators may detect the payload deployment and not the launcher itself. To make detection even more difficult, they usually rely on nontraditional techniques to hide their activity [19].

5. Information-stealing Malware

This malware collects information based on its design and sends off the infected computer in a manner that enables the attacker to retrieve it. A keylogging malware will capture any and all characters typed on the associated keyboard. A password grabber scans the computer for hashes stored locally when a user selects to be remembered by his bank’s website while logging into his banking account.

6. Backdoor

A backdoor bypasses security mechanisms allowing uncontested access to a computer program. Sometimes programmers put them in place to facilitate troubleshooting faulty software. Attackers either exploit existing backdoors once discovered or put them in place to facilitate uncontested access to a workstation on the network. From the victim workstation, an attacker can stage additional attacks on other network machines difficult to execute external to the network since most security programs protect the network from unauthenticated, external access. Backdoors, sometimes referred to as "trap doors," facilitate internal threats to a network's security. If machines infected by a backdoor receive commands from the same command and control server they form a botnet [19]. The infected machines are called bots or zombies.

7. Rootkit

Root kits hide in and exploit the base functionality of the host operating system in order to facilitate control of certain aspects of that operating system while remaining hidden [21]. Rootkits are installed once an attacker has gained access to the machine and has found a way to exploit the authentication process allowing him to install the root kit as an administrator. It uses the administrator or root level privileges to take actions to hide itself and other malware. Put simply, it is a "kit" that allows the attacker to establish and retain "root" access [21].

D. MALICIOUS ACTIVITY

Malware enables an attacker to compromise the confidentiality, integrity, and availability of computer networks, systems, or information contained within. Although malware may be unintentionally deployed on a system, the design of malware is intentional. Its maker had specific intent for it when considering its design. Some programs are not intentionally malicious. How they are used may be malicious. As an example, "rm –r *.*" is an example of a misuse of the useful UNIX command remove, or "rm." Like the variants of the types of malware discussed above, there are many ways an unauthorized person may gain access to a computer. Once on the computer, the person is inside the castle walls. Like a castle, the bulk of a network's defenses focus on preventing

access to unauthorized persons. Many defense-in-depth security architectures are designed with the assumption that any activity on the inside of the network's defenses is legitimate and, therefore, warrants less monitoring or restraints. The methodology with which an attacker advances his attack is described in the Certified Ethical Hacking Manual [22]. The phases of an attack are:

- Reconnaissance
- Scanning
- Gaining access
- Maintaining access
- Clearing tracks

By the nature of its design, malware will enable one or more of these phases. A malware that conducts a port scan would be employed during the scanning phase. During the reconnaissance phase, a socially engineering attack in the form of an e-mail from a legitimate source may contain a virus or a hyper-link to a malicious website containing drive-by download malware, a type of downloader. Both an e-mail attachment and malicious scripts hidden in a website may be used to enable the attacker's efforts in the gaining access phase. During the clearing tracks phase, a malicious pop-up window may have a script tied to its "Close" button that launches an evidence-erasing virus when selected by the user.

1. Port Scans

Computers send and receive traffic based upon standards and agreements made during session establishment. Both the sender and receiver of network traffic identify the port numbers from which data will be transmitted from and to. The port numbers associated with a transmission act as mailing addresses for network traffic. An e-mail sent from a mail server to a work station will be directed at Port 25, which is dedicated to Simple Mail Transfer Protocol traffic. When someone uses a web browser to reach the local newspaper's website, their computer establishes a connection with the web server hosting the website via Port 80, Hypertext Terminal Protocol (HTTP). Port scanning software has a legitimate purpose. It is used by network administrators to verify the network is in compliance with the network security policy as well as for checking service

availability. It also may be used by attackers during their scanning phase to identify open ports or services offered by a targeted network. Equipped with target Internet Protocol (IP) addresses gained during the reconnaissance phase, the attackers exploit the functionality the Transmission Control Protocol (TCP) [23]. A TCP segment has flags that provide guidance during the establishment, management of, and disestablishment of a session between sender and receiver. The flags are abbreviated URG for "Urgent," ACK for "Acknowledge," PSH for "Push," RST for "Reset," SYN for a "Synchronize," and FIN for "Finish." A network compliant with the TCP specification (RFC 793) may be exploited into divulging which ports in its firewall are open and which are closed [23]. Programmers have developed software programs like NMAP (Network Mapping Protocol) that automate port scanning. Port scanning is the process of sending segments to a target IP address to multiple ports and with different combinations of TCP flags. The information returned allows the attacker to guess what operating system is running on the target machine as well as what services (e.g., web server, File Transfer Protocol (FTP) server, mail server) are running on the target machine. With this information, the attacker identifies exploits of the operating system and software used in providing these network services in order to select the tools he will use to conduct his attack.

2. Social Engineering Attacks

An attacker conducting a social engineering attack is utilizing his social skills and knowledge of human interaction to manipulate the target into doing what he wants. In this type of attack the target is a human and not a workstation or network. The goal of most social engineering attacks is to capture credentials tied to user's personality, lines of credit, or banking accounts [11].

A social engineering attack may also convince a person to click on an e-mail's embedded hyperlink or to open an attachment. Most mail servers are configured to block e-mail attachments that are executable files. Anti-virus programs play another role in preventing the delivery of attachments that match a signature of a known malware. A hyperlink is not an attachment. Again, a properly configured mail server can remove hyperlinks from e-mails before they are delivered. The end user is not defenseless in the

absence of this mail server configuration. Users may change the options of their mail program to convert all e-mails to text only. Regardless, malicious attachments and hyperlinks may still get through these security measures and arrive in a target's inbox. Also, a user may select to copy the uniform resource locator (URL) of the malicious website from the socially engineered e-mail and paste it into the address bar of their web browser. Once opened, the attachment may be a Trojan horse, a downloader, or an information-stealing malware depending on what the attacker requires to advance his attack to the next phase.

Malicious links may direct the user to malicious websites containing malicious java script enabling the attacker to further compromise the security of the user's workstation. The malicious website may contain false links that fool the user into downloading malware. The hyperlinks might even take the user to legitimate websites whose web-hosting servers have been compromised. For example, attackers wishing to hide their malicious content from web administrators may use zero-pixel iframes. An iframe, or inline frame, is used when the web designer wants to embed a separate document within the current web page [24]. A zero-pixel iframe is too small to see. The iframe, injected at the compromised web-hosting server, may contain malicious java script that may manipulate the victim web browser to visit other malicious sites designed by the attacker containing additional malware. The intent is to re-direct the user's legitimate Internet traffic to these malicious sites where he will be persuaded to download malware that gives the attacker access to the machine and, subsequently, the attached network.

Malicious pop-up windows are another vehicle of social engineering attacks. There are various reasons for pop-up windows. A network administrator may use a pop-up window to send notice to users that an emergent system reboot is required to finish the installation of a software update. Some system reboot notifications come with radio buttons providing the user options such as "Restart Now," "Delay Restart," and "Cancel." websites use pop-up windows to provide advertisements to the user. The designer of a malicious pop-up window will tie malicious code to the radio buttons. When the user clicks on a malicious radio button within the pop-up window, malicious java script

downloads malware to the victim machine providing the same control to the attacker as mentioned above [25].

E. CYBER SECURITY INTIATIVES IN THE DOD

1. Dynamic Approach to Network Defense

In the 2010 Quadrennial Defense Review (QDR) report, then Secretary of Defense (SECDEF) Robert Gates set the course toward improving the readiness of DoD networks to operate in cyberspace by identifying a need for greater capability to defend against attacks in cyberspace. SECDEF Gates underlined a demand for improvement in existing capabilities to counter threats existing in cyberspace [26]. The threats to the confidentiality, integrity, and availability of DoD networks posed significant potential to hinder the DoD's ability to execute its missions. This guidance from the office of the SECDEF led to the DoD identifying cyberspace as a warfare domain. U.S. Cyber Command (USCYBERCOM) was established and assigned the mission of ensuring the secure use of cyberspace by the DoD for the execution of U.S. National Policy. USCYBERCOM's Concept of Operations (CONOPS) identified Lines of Operations (LOO): DoD Global Information Grid (GIG) Operations (DGO), Defensive Cyber Operations (DCO), and Offensive Cyber Operations (OCO) [27]. The portion of USCYBERCOM's mission that was not receiving due diligence was ensuring the operations of the GIG. DGO is the USCYBERCOM LOO responsible for ensuring continued operations of the GIG. DGO covers the creation, maintenance, and protection of all DoD networks [27]. Whereas DCO provides for the defense of DoD networks and their access to and use of the GIG, DGO provides protection in the form of self-defense.

To date, existing self-defense operations include ensuring a network is equipped with properly configured firewalls, running anti-virus applications with up-to-date anti-virus signatures, operating with the latest vendor-supplied updates for the operating system and applications running on the network, and ensuring no unauthorized applications are running on the network. This defense-in-depth concept assumes that, once in place, the overlapping "layers" will ensure the safety of the network from malicious activity and malware. It is a static approach to providing self-defense against a

dynamic threat. Not only is the network and its information dynamic but the threats are also dynamic. In his 2001 article "Learning to Operate in Cyberspace," Rear Admiral (Upper Half) William E. Leigher, Director of Warfare Integration for Information Dominance (OPNAV N2/N6F) [1], captured the direness of the state of DGO efforts in the DoD. He expressed concern for current objectives of operations in cyberspace stating such objectives would not be accepted in any of the other warfare domains [1]. RDML Leigher noted that the effort to provide a well-fortified network does not prevent all attacks. A dynamic approach to self-defense is required to address a dynamic threat. Shipboard damage control readiness provides an example of how to dynamically approach network self-defense shortfalls.

2. Casualty Responsiveness on the Network

The threat to a U.S. Navy ship's operational readiness from fire or flooding is great. Upon entering service in the U.S. Navy, enlisted and officer ranks are provided initial shipboard damage control and firefighting training. Both enlisted and officer alike learn how to respond to fire and flooding casualties. They gain cursory knowledge of their casualty-fighting equipment and procedures. These tried and true casualty response procedures and methodologies are developed from lessons learned by real crews facing actual fire and flooding casualties. The training scenarios are well-planned to ensure realistic simulation of actual casualties experienced in the Fleet. To ensure a realistic simulation, the training environments are designed to closely reflect operational spaces in which the training audience might encounter fire and flooding while at sea. The skills gained during initial training are re-enforced when assigned to a Navy ship. Through continued training, these crew members, when assigned to the ship's damage control or firefighting teams, gain a familiarity with their equipment, their procedures, and their ship. They develop procedural memory, or "muscle-memory," through repetitive damage control and firefighting drills. This "muscle-memory" is vital to ensuring fast and effective response of the damage control and firefighting teams. Familiar with their equipment and well-exercised in their casualty response procedures, these teams are empowered to gain control of a casualty and quickly return the ship to its full operational capacity.

Like a ship's crew ensuring its readiness to respond to a fire or flooding casualty, network administrators require an avenue with which to re-enforce and further develop network casualty response skills while also developing a familiarity with their network. They need to learn the strengths and weaknesses of themselves and those of their fellow network administrators. The threats faced by DoD networks were discussed in Chapter II. Some of these threats fall outside of the scope of network self-defense and require the capabilities provided by the DCO mission. Some threats do fall within the means of a network administrator to detect, either by human observation or detection by software or hardware readily available to them. It is the detectable threat that DoD network administrators need the opportunity to face in a simulated environment mirroring the operational networks they are responsible to create, maintain, and protect. Having the opportunity to develop and practice their cyber casualty response trade craft establishes a higher probability that network administrators will successfully combat actual threats and the resultant damage sustained by the network. The DoD utilize red teams, groups of ethical hackers, to run simulated attacks on DoD networks in order to assess the cyber readiness of its networks and administrators. Unfortunately, these assessments are performed, at best, once or twice a year. In addition, there are no training sessions of any kind in which cyber threat simulation is provided to allow network administrators the opportunity to hone network casualty response and damage control skills. In fact, most techniques and procedures, if in existence, are home-grown and not doctrinal in nature. The interval between training opportunities is too great for any skill to be effectively developed and retained.

Red teams are not the solution. The DoD needs a cyber damage control training environment. Red team assessments of command network defense are too infrequent for the network administrators to experience the variety and repetition of attacks necessary to develop the familiarity and skills required of them. This infrequency is due to funding constraints, which limit the DoD red team growth in a time when the demand signal for cyber assessments and training continues to increase. More importantly, creating a training environment is not the purpose of a red team. They are an inspection entity

responsible for assessing the cyber readiness of a network and the ability of its network administrators to provide self-defense.

What is required is a cyber damage control training tool capable of simulating real cyber casualties while safely providing this simulation without threat to the operational readiness of the host network. This cyber damage control trainer should be readily available at the unit level to facilitate training opportunities as frequent as the unit level commander deems necessary and at times of his or her choosing.

F. AN OVERVIEW OF MAST

MAST aims to deliver scalable, safe, and realistic simulation of malicious activity to network administrators and operators. MAST is an excellent tool for developing, improving upon, and assessing cyber readiness of its target audience. Its simulations may be utilized to conduct initial malicious activity recognition and response training, to re-enforce that training through repetitive exposure, or to assess the effectiveness of that training. The target audience may be the network administrators who will act as first responders to an incident of malicious activity on the network. It may also be the network users who provide for a greater security posture of the network if enabled to recognize malicious activity on their workstations or even prevent potentially malicious activity through thoughtful use of their workstations and access to the Internet. As shown in Figure 1, the MAST Network consists of two parts: a Scenario Execution (SE) Server hosted by the local network and a Scenario Generation (SG) Server. This figure depicts the ability of MAST to remotely configure, initiate, monitor, and terminate Simware modules on the host network of the simulation audience. Network administrators and local network users may receive MAST-aided training and assessment on any LAN regardless of the geographical location of the participating network. Whether at an established headquarters, a forward deploying unit, or on a ship underway, access to the GIG provides the SG Server with connectivity to the local SE Servers. From the SG Server, the evaluator verifies participating SE Servers are on-line and pushes any revisions or additions to Simware module libraries maintained on the local SE Servers. Execution of the Simware module is ordered by the evaluator through the SG Server and

carried out locally by the SE Servers. Status of the simulation on each local network is monitored by the local safety observers through information retrieved by the SE Server. The same information is relayed to the evaluator at the remote location through the SG – SE Server command and control channel. All data is retained and archived by the SE Servers. Further detail regarding the individual components of MAST is provided next.

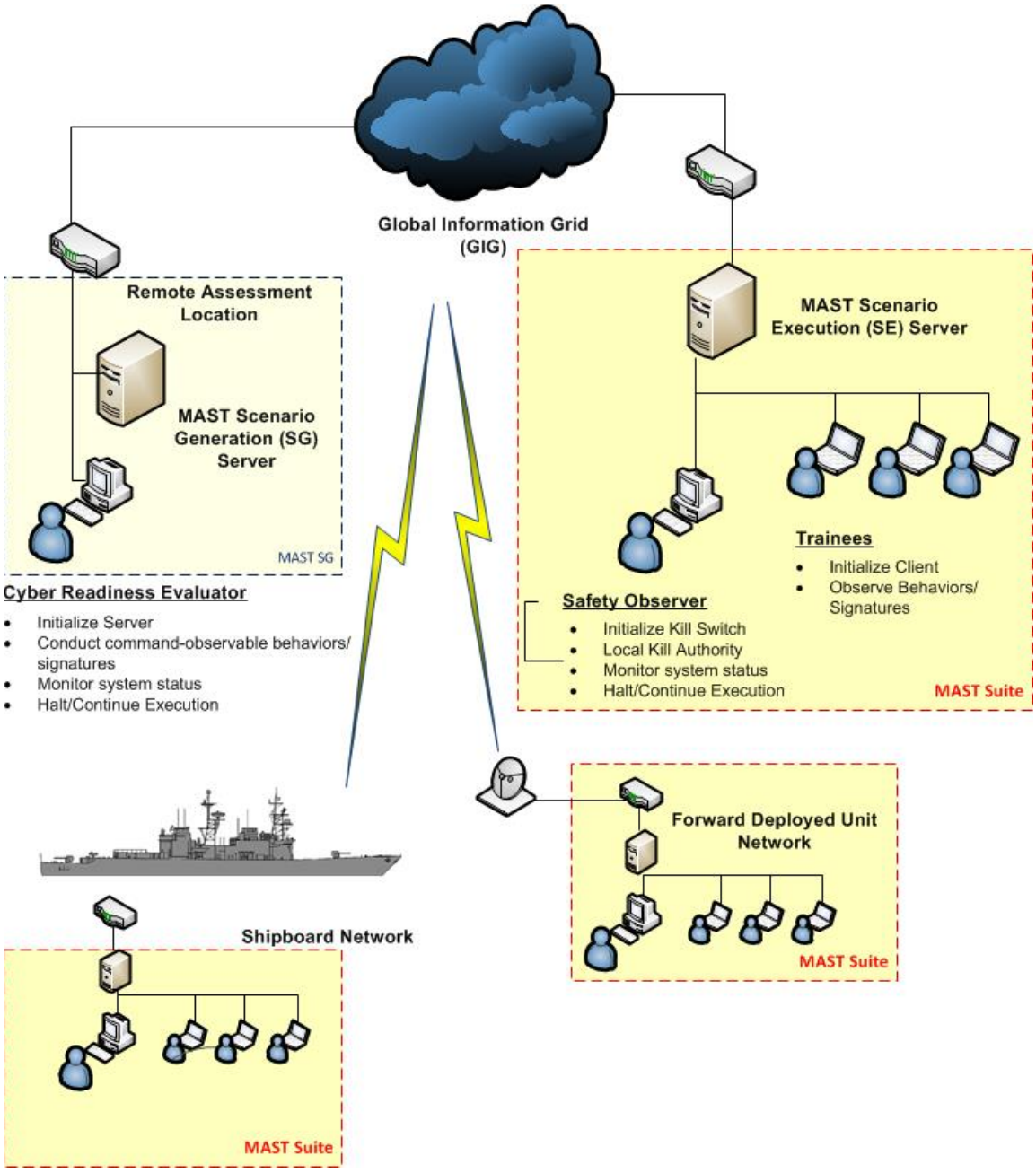


Figure 1. MAST Architecture Overview. From [3].

1. The MAST Suite

MAST provides the unit commander with the ability to develop and test the skills to recognize and respond to cyber attacks in unit network administrators. It also creates an environment that lends to development and modification of until-level, cyber attack casualty response procedures. MAST is constructed as a server-client architecture. It consists of a SE Server, a database server that collects all associated data, and the client applications of the MAST software loaded to each network workstation participating in the simulation. The SE Server contains and executes the Simware modules. It provides a graphical user interface (GUI), which equips the simulation master with functionality necessary to select which Simware to execute and to control the execution of that Simware. The participating audience is determined by the installation of the MAST client application on a workstation. The Simware, once executed, will only run its course on workstations connected to the same network as the SE Server and running the MAST client application. The MAST suite may be considered the "MAST-light" system. Although not fully the MAST system, it is an autonomous entity able to execute scalable, safety, and realistic simulation of malware and malicious activity [4].

2. The Scenario Generation Server

Although very much a part of the MAST system, the SG Server is considered its own entity whose functionality may be divorced from the rest of the MAST system without affecting the system's ability to execute its core capabilities. The SG Server provides the ability to dictate the functionality of SE Servers as part of its parent-child relationship established remotely through network connectivity with all participating SE Servers. In addition, the SG Server may delete, add to, or modify existing Simware modules in Simware module libraries of any SE Server it is connected to. The network connectivity also allows remote monitoring of Simware execution conducted by each connected SE Server. This combined functionality completes the MAST system by providing a single, remote location with the ability to safely execute and control [4]. Since our research scope focuses specifically on the network hosting the simulation, we

will only be testing the MAST compatibility with our test LANs running the COMPOSE environment. In our testing of MAST, we will not including the SG Server functionality.

G. PROOF OF CONCEPT FOR A MALICIOUS ACTIVITY SIMULATION TOOL

Hayes produced a summary of the three waves of thesis research projects associated with the MAST program in Chapter II of his thesis, "A Definitive Interoperability Test Methodology for the Malicious Activity Simulation Tool (MAST)" [6].

- First Wave: Taff and Salveski
- Second Wave: Neff/Longoria
- Third Wave: Hayes/Makhlouf and Littlejohn

The foundation for this project was laid by Taff and Salveski. Their research established that a software system could be created to mimic malicious activity on a network without actually deploying malware on the network. Their thesis, "Malware Mimics for Network Security Assessment," described the concept of utilizing simulated adversarial teams to attack networks and how their research software system could provide an equivalent training environment. The characteristics of the system constructed by Taff and Salveski were [2]:

- It was safe enough for production or operational environments. Emulated behavior would cease on command and "roll back" to a pre-exercise state. Losses of network connection were treated as instructions to cease behaviors. This would allow training to take place on the same network on which the trainees perform their mission.
- Only malware behaviors were constructed, not actual malware itself. Though (they) demonstrated the properties of a notional worm on the network, there was no actual malware involved.
- The system constructed was distributed across the network, allowing the trainer to be located anywhere on the network, local or remote.

Neff followed Taff and Salveski's research by verifying and validating their proposal that their malware mimic software system, which Neff re-labels as the MAST system, is a viable tool for conducting network security training. Neff described various attack methods used by network security training teams during network attack simulations. He defined and tested metrics, which he used to compare the MAST with other DoD network security training tools [3]. He concluded that the MAST system was capable of producing equal training quality to that of the training methodologies used today. In addition, he explained that the MAST system was capable of presenting real world simulation through updated malware modules shaped to represent present day network attacks. Longoria's research assessed the MAST Software's ability to be remotely deployed to networks of varying sizes with little to no impact on the network or its resources [4]. His research demonstrated the MAST software could be deployed to as many as 80 workstations on a network without placing a significant load on the Simulation Execution (SE) server's central processing unit (CPU). In addition, the overall network traffic increase due to Simware deployment was minimal. Additionally, Longoria showed that the MAST clients installed on each workstation participating in the training simulation successfully transmitted its reports to the SE server. He concluded that each instance of the MAST client software reported all user actions and pertinent local machine information to the SE server as they occurred [4]. Transmission of user action reports from both the clients to the SE server placed minimal additional burden on the transmitting machine's CPU.

Hayes' research laid the ground work for the quantitative testing we will perform in our research. Hayes captured the current DoD and DON testing and implementation methodologies for cyber programs [6]. Specifically, Hayes identified testing objectives, milestones, and metrics as well as the methodology for a quantitative testing of MAST. He shaped and developed test procedures that, if followed, would test and aid the evaluation of a software program in order to assure its readiness.

H. SUMMARY

In this chapter, we discussed the nature and employment of malware. We identified the tenets of information security: Confidentiality, Integrity, and Availability. We explained how something or someone can pose as a threat to a network, its systems, or the information within through attacking one of several of these security objectives. We also explained that most threat agents fall into one of three groups: criminals, hacktivists, or government/government-sponsored agents. We provided a definition of malware and described seven different categories of malware. We discussed how a criminal, hacktivist, or government/government-sponsored attacker would conduct malicious activity to enable their attacks on systems. We described some of the cyber security initiatives in the DoD. Finally, we provided an overview of MAST.

In the next chapter, we will describe the concept behind the implementation of the MAST program as a training and assessment tool. We will also include a brief overview of the MAST program and the COMPOSE environment.

THIS PAGE INTENTIONALLY LEFT BLANK

III. DESIGN CONSIDERATIONS

MAST is designed to provide realistic, tailored simulation of malicious activity to facilitate effective cyber readiness. To create a realistic simulation scenario requires three factors:

- Simulation executed in a real computing environment
- Simulation of real malicious activity
- Safety and scalability

In the first factor, "real" refers to an operational computing environment in which the simulation audience may encounter malware or malicious activity. In the second factor, "real" refers to malware or malicious activity actually present in the wilds of the Internet. The types of malware and malicious activity discussed in Chapter II reflect real malware and real malicious activity. The final factor is also essential to a simulation. Without the ability to safely confine the execution of a simulated malware or malicious activity to the time and manner desired, it is erroneous to assume a "simulation" could not potentially become an actual occurrence of malware or malicious activity. Also, the realistic effect of a simulation is hampered if the simulation cannot scale to meet the needs of its audience or environment. This chapter describes how these factors are accounted for in the COMPOSE testing environments and the MAST Simware.

A. PROVIDING A DON COMPUTING ENVIRONMENT

As mentioned in Chapter II, previous research projects proved the functionality of MAST as well as its scalability while operating in a virtual COMPOSE environment. Our research moves the project forward by assessing the MAST system's ability to execute all functions as designed while running in a COMPOSE environment on a physical network.

We decided to execute our testing in two separate environments. One environment would be a MAST Test Bed built at the Naval Postgraduate School (NPS). The other would be the NCOR in Norfolk, VA. Testing MAST on the MAST Test Bed meets the intent of our research. Even so, a successful test and evaluation of the MAST functionality on the MAST Test Bed does not negate the need to test MAST on a

COMPOSE network certified and accredited to access the GIG. Testing MAST and its Simware on the MAST Test Bed would provide a clear indication of its compatibility with our instance of a shipboard-simulating COMPOSE environment. Testing of MAST on NCOR would give our results the authenticity that only comes from operational testing on a certified and accredited network.

1. The MAST Test Bed Environment

The MAST Test Bed that models a shipboard network did not exist at NPS at the beginning of this research project. So we constructed and configured a physical network in our research laboratory space. The MAST Test Bed needed to model the operational requirements of a shipboard network. It was designed for the sole purpose of replicating, not mirroring, a properly configured, operational Navy shipboard network.

Considering that the scalability of the MAST system was proven by Longoria through his research, it was unnecessary to match the capacity or complexity of any Navy shipboard network [4]. Its anti-virus software, intrusion prevention program and firewall for the network and the individual workstations are configuration in accordance with DoD and DON policies. This was very important to the success of our research because it is the intent of MAST to create test modules tailored to produce the desired response when run on an operational network. While running a Simware module scenario, an error in indications or functionality would point to either an error in the module's design or a misconfiguration in the COMPOSE network. Unlike an operational network, we did not pursue GIG-connectivity for the MAST Test Bed. Next, we describe the hardware and software used in creating the MAST Test Bed. Table 1 provides a summary and maps hardware to software:

Server Name	Server Function
MASTUCSDC001 *Windows Server 2003*	Domain Controller Global Catalog Server File and Print Services COMPOSE Distribution Server Primary Windows Internet Naming Service (WINS) Primary Dynamic Host Configuration Protocol (DHCP) Primary Domain Name System (DNS)
MASTUCSDC002 *Windows Server 2003*	Domain Controller Global Catalog Server File and Print Services Internet Security and Acceleration Server (ISA) Secondary Windows Internet Naming Service (WINS) Secondary Dynamic Host Configuration Protocol (DHCP) Secondary Domain Name System (DNS)
MASTUCSEX001 *Windows Server 2003*	Microsoft Exchange 2003 Primary Server Internet Information Server (IIS) Web Server SharePoint Server Network News Transfer Protocol (NNTP) Server
MASTUCSEX002 *Windows Server 2003*	Microsoft Exchange 2003 Secondary Server Internet Information Server (IIS) Network News Transfer Protocol (NNTP) Server
MASTUCSMS001 *Windows Server 2003*	SQL Server 2005 Standard Internet Information Server (IIS) SMS Distribution Point Systems Management Server (SMS) Remote Installation Services (RIS) Server
MASTUCSFS001 *Windows Server 2003*	Profile Server Home Directory Server
MASTCNDOSE *ESXi*	Host Based Security System (HBSS) *Microsoft Windows Server 2003*
	Secure Configuration Compliance Validation Initiative (SCCVI)
	Microsoft SQL Server (MSSQL) *Microsoft Windows Server 2003*

Table 1. MAST Test Bed Hardware and Software

a. Hardware

The primary requirement for our test environments is the existence of the COMPOSE environment deployed on U.S. Navy ships. Deployment of the COMPOSE environment requires four basic components: Servers, a router, switches, and workstations. The requirements for the MAST Test Bed did not exceed this level of specificity, so no great effort was placed in hardware selection above ensuring we possessed these basic network building blocks. The NPS Test Bed was constructed utilizing pre-existing equipment in the Computer Science Department reserves. Due to practical reasons, we opted to use six physical workstations. Our Servers, router, and switch were stored in a Dell PowerEdge 4220 Rack Enclosure. The hardware specifications of our equipment are as follows:

- **Server (7)**: Dell™ Poweredge™ 1950 Server (2 x 2.66 GHz, Quad-Core Intel® Xeon® 5300 processors; 12 Gb, Fully-Buffered DIMMs, 4 x 500 Gb 3.5" SATA hard drives.)
- **Router (1)**: Cisco® 2811 Integrated Service Router
- **Switch (2)**: Dell™ PowerConnect™ 2716 (16 x 1-Gigabet Ethernet ports)
- **Workstation (6)**: Dell™ OptiPlex™ GX620 Ultra Small Form (2.8 GHz Intel® Pentium® 4 processor with Intel® 945 Express support chip; 2 Gb 533MHz, DDR2 memory; 250 Gb hard drive)

The network topology depicted in Figure 2 shows a connectivity to the Internet through the NPS network. This connection was only a temporary one used to download software and firmware upgrades required during the software installation and configuration process. Once finished, the connection was removed, ensuring an isolated environment to run the MAST Simware modules. The second switch (192.168.100.x), attached to the Test Bed router, allows for the simulation of an external network. This external network, still isolated from the NPS network, and therefore the Internet, facilitates future testing of the SG Server and its remote control of the MAST Suite. In addition, the external network provides the ability to simulate malicious activity initiated external to the internal (192.168.10.xxx) network hosting the MAST Suite.

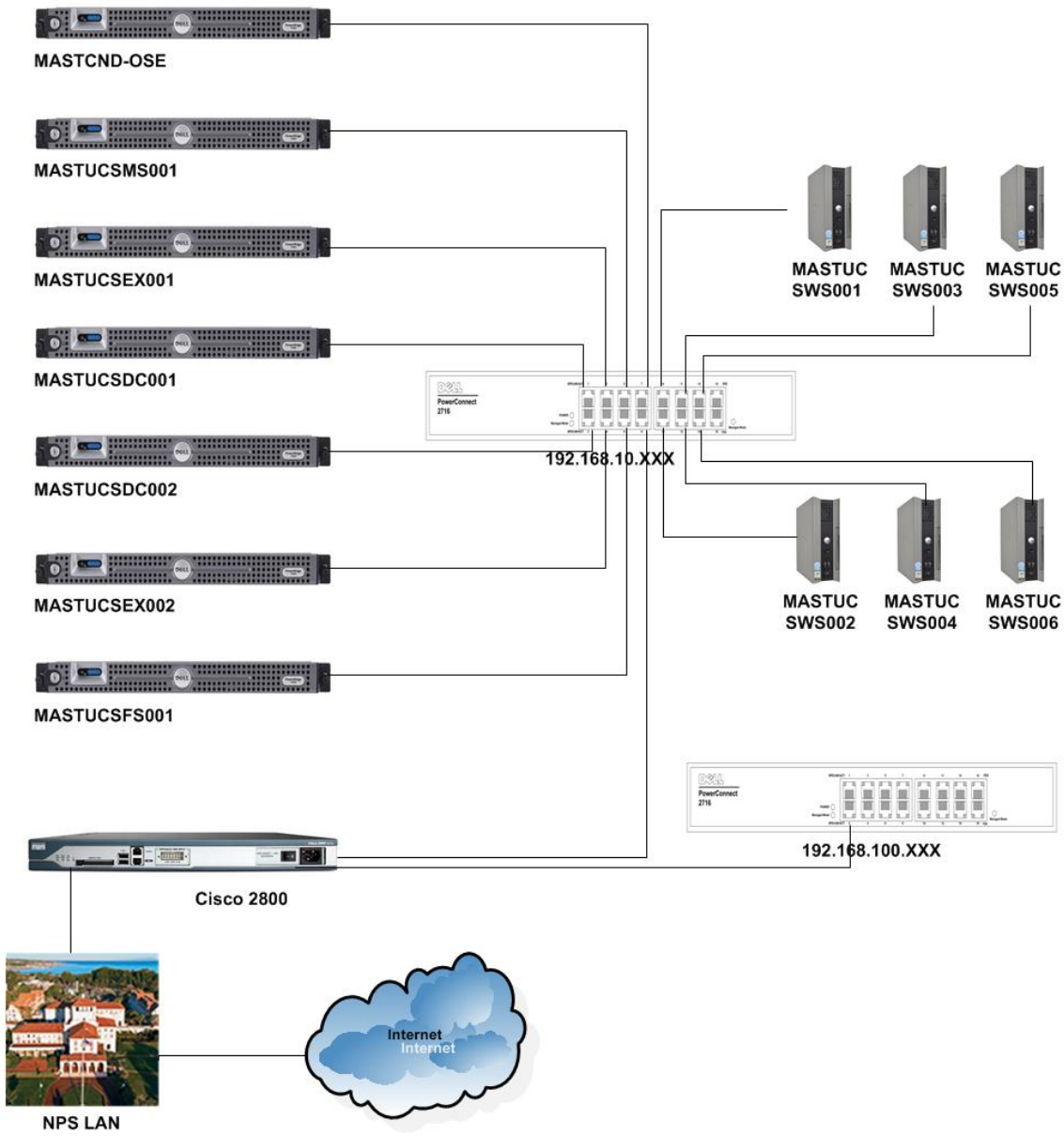


Figure 2. NPS MAST Test Bed Network Topology with Nomenclature

b. Software

The software required to emulate a shipboard COMPOSE network is grouped into packages, which we will refer to as loads. We required two loads to properly establish the NPS Test Bed. The first COMPOSE load contains the software needed to create the network. The COMPOSE load, version 3.5, was donated by the Space and Naval Warfare Systems Command (SPAWAR) Program Office for Tactical

Networks (PMW-160). The second load delivers the Computer Network Defense-Operating System Environment (CND-OSE), which includes the DoD-mandated Intrusion Prevention System (IPS), the Host-Based Security System (HBSS). HBSS is a McAfee® intrusion prevention and policy management software. It detects, and prevents known cyber-threats and is required to be installed on each server and client in DoD. CND-OSE, version 2.0, was donated by the SPAWAR Program Office for Information Assurance and Cyber Security (PMW-130). Both were provided with 120-day operating licenses. All servers were configured with a 20-Gigabyte "C" drive system partition and 230-Gigabyte "D" drive partition. Both partitions were configured as 32-bit NFTS. Microsoft® Windows Server® 2003 R2 Standard was installed on the six servers implementing the COMPOSE load. The seventh server utilized VMware ESXi™ implement the CND-OSE load. ESXi™ is a hypervisor that creates and controls virtual connections between the three CND-OSE Servers to the physical memory and processors of the host server. The use of a virtual environment to run the CND-OSE does not negate the physical LAN requirement of our testing. All CND-OSE suites deployed on board U.S. Navy ships utilize the same configuration for its three servers.

(1) MAST Test Bed Domain Controllers . MASTUCSDC001 and MASTUCSDC002 function as the domain controllers for the COMPOSE network. Both DC001 and DC002 functioned as complimenting domain controllers. DC001 provided the following additional service: COMPOSE Distribution Server, Global Catalog Server, Primary Windows Internet Naming Service (WINS) Server, Primary Dynamic Host Configuration Protocol (DHCP) Server, Primary Domain Name System (DNS) Server, and Symantec Anti-Virus™ Server. DC002 provided the secondary WINS, DHCP, and DNS Servers.

(2) MAST Test Bed Exchange Servers. Just as with domain controllers, COMPOSE environments provide redundancy to the important role of the network's exchange servers. MASTUCSEX001 and MASTUCSEX002 function as the exchange servers. Both host the Microsoft® Exchange 2003 Server® Standard Edition and Microsoft® Internet Information Server (IIS). EX001 also hosts the network's web server and Microsoft® SharePoint Server®.

(3) MAST Test Bed Microsoft® Mission Management Server®. MASTSMS001's primary function is the Microsoft® Mission Management Server® (MMS). It is also loaded with Microsoft® SQL Server 2005 Standard, IIS, and the SMS software distribution server.

(4) MAST Test Bed File Server. MASTUCSFS001 functions as the network file server as well as its home directory server. It also maintains the share drive and all user profiles.

(5) MAST Test Bed CND-OSE Server. As mentioned previously, this physical server, running VMware ESXi™, hosts the virtualized environment for the CND-OSE suite. The CND-OSE suite is virtually implemented to isolate its security functionality from the rest of the COMPOSE environment. The following four, virtualized servers comprise the CND-OSE suite: the Host-Based Security System (HBSS), the Microsoft® SQL Server® (MSSQL), the Secure Configuration Compliance Verification Initiative (SCCVI), and the Secure Configuration Remediation Initiative (SCRI). The full CND-OSE suite was loaded to the server as a measure of thoroughness. In previous works, Lieutenant Commander Neff and Captain Longoria provided detailed descriptions of the HBSS and its role in the COMPOSE environment [3], [4]. Only the HBSS suite is required by our testing environment criteria. The HBSS suite is actually a collection of McAfee network security products managed by the e-Policy Orchestrator (ePO) application loaded on the HBSS Server. From the ePO, McAfee ePO and Host-based Intrusion Prevention System (HIPS) agent applications were deployed to the Test Bed Servers and workstations. Through the interconnections between the ePO and the individual McAfee ePO and HIPS agents, we pushed DoD and DON security policies to the Test Bed Servers and workstations. These policies were applied to the servers and workstations by the hosted agents. The agents networked servers and workstations are configured to poll the ePO in pre-determined time intervals for any new policies or updates to existing policies. If any are received, the agents apply them to their host machine. Any detected violations of applied policies are reported by the ePO and HIPS agents both locally for host user situational awareness but also remotely to the ePO console through the existing interconnection. In order to ensure

resilience in the ability to prevent, detect, track, and report malicious activities on the network, the CND-OSE installation instructions do not direct the deployment of the McAfee anti-virus agent to the individual servers and workstations. Instead, COMPOSE networks utilize Symantec's enterprise anti-virus solution. As mentioned earlier, Symantec Anti-Virus™ Server is loaded on MASTUCSDC001. Like the ePO, the anti-virus server possesses the ability to push both anti-virus agents, configuration policies, and policy updates to all networked assets. Also like the ePO and HIPS agents, the anti-virus agents communicate alerts and warning locally and remotely to the anti-virus server. This feature allows the host machines to act as sensors for the both the ePO and anti-virus servers. Between the two, the network administrators and security personnel have the mechanisms necessary to maintain situational awareness of the cyber security posture of their networks as well as detect and respond to any malicious activity. The combined functionality provided by the HBSS and Symantec Anti-Virus™ Server installations on the MAST Test Bed is also provided in the NCOR testing environment, which will be described next.

2. The NCOR Environment

Of the cyber test ranges researched and recommended by Lieutenant Hayes, we selected to conduct part of our research-related testing on the NCOR [6]. As mentioned in Hayes's work, NCOR provides DON with an arena of computing and networking capability in an isolated environment. This isolation allows for freedom to conduct cyber training, exercise, and test and evaluation events that might otherwise pose a threat to operational or production networks. Utilizing the COMPOSE environment, this arena allows the engineers the flexibility to change the stage to meet the needs of its customers. The majority of the time its customers are Navy network defenders. They utilize NCOR to provide them with a cyber dojo in which the trainers show the students attacks at a digestible pace. In this dojo, the students learn to recognize attack vectors and how to defend against these attacks. Their trainers can repeat the attack as many times as necessary for the students to recognize the attack and instinctively deploy the defense. This enables the reinforcement of their training, the honing of their recognition and response skills, and even aids in the development of unit camaraderie. The same

flexibility needed to change or remove the NCOR defense-in-depth layers as necessary for training also allows for the creation of whatever stage is needed to conduct testing and evaluation events. Whether the test subject is a human or a computer program, the evaluators can work with NCOR engineers to adjust its simulated shipboard network configuration as necessary to create the desired environment. Given the commonality between NCOR's ability to provide a cyber dojo and MAST's aim to make a cyber dojo available at the unit level, plus the fact that the NCOR engineers use and are familiar with the COMPOSE environment, NCOR is the natural fit for our secondary test environment. Of the simulated domains provided by the NCOR engineers, only the CVN-65 domain met our testing criteria of a fully-configured COMPOSE environment integrated with the HBSS-suite [28]. The CVN-65 domain's COMPOSE load was version 3.0.1, an older version than the MAST Test Bed's 3.5 COMPOSE load. This disparity was assessed and deemed not an eliminating factor.

B. SIMULATING MALWARE AND MALICIOUS ACTIVITY

When designing simulations of malware and malicious activity, the designer has to know the capabilities of the target audience and the operational network he is trying to replicate. In the interest of fiscal judiciousness, it is an unsound practice to train a network administrator, whether in the private sector or in the military, on how to respond to malicious behavior that is beyond the scope of his capabilities to detect or the capabilities of the network security defenses with which the administrator is equipped to defend his network. In his research, Hammond suggests that malicious behavior may be viewed as falling into one of four categories as depicted in Figure 3 [5].

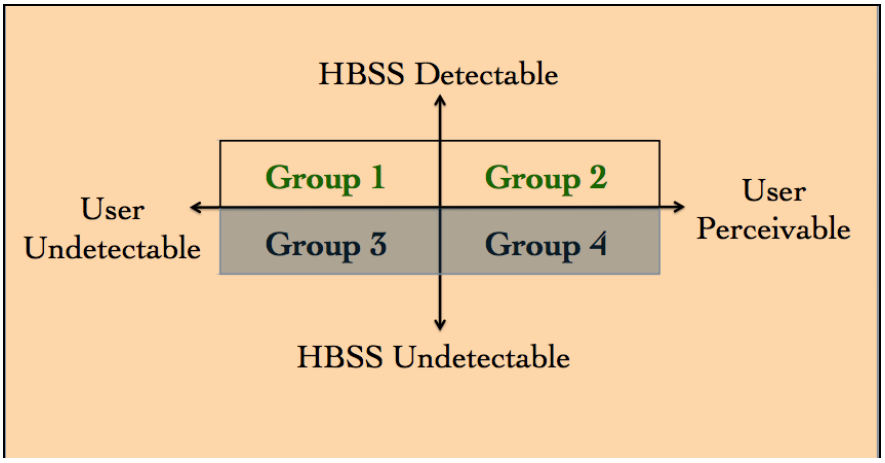


Figure 3. Malicious Behavior Detection Diagram. From [5].

Hammond offers that any malicious behavior on a network should be treated as part of a group. The grouping is determined by whether or not the malicious behavior is detectable by a user (to include network administrators) and whether or not the behavior is detectable by network security features, such as the HBSS. Taking this into account, we designed our Simware modules to replicate malicious behaviors that only fall into Groups 1 and 2. Although not denying the existence of the significant volume of Group 3 and 4 malware and malicious activities, the intent of the MAST project is to generate a realistic, and therefore beneficial, training tool. Simulation of Group 3 and 4 malicious behavior is of no training value and falls outside of the scope of this research.

1. Developing Training Scenarios

Existing Simware modules were created for the purpose of testing the MAST system's functionality. Viruses and drive-by downloads are examples of malicious activity still posing a threat to networks worldwide. Previous MAST research projects successfully developed and utilized Simware modules that present the simulation audience with an accurate simulation of a malicious behavior. In addition to replicating malicious behavior, the Simware modules facilitate dynamic evolution allowing for secure or insecure audience reaction to the simulation. Behind the scenes of the interaction between the module execution and the simulation audience, the module design also supports reporting and archiving of data associated with the module execution. See

Longoria [4] for a more thorough explanation of the features and functionality of a MAST Simware module. Figure 4 depicts how the MAST system executes the simulation of a drive by download [4].

In our research, we focused on the potential training scenario a unit would face when participating in a MAST training evolution. Our training scenario enables the assessment of user compliance with network usage policies and the effectiveness of DoD-mandated, annual information systems security awareness training. The Simware modules created to deliver a training scenario provide the trainer or assessor the flexibility to utilize individual Simware modules to focus on the simulation audience's response to one particular malicious activity or combine Simware modules to present a more complex evolution.

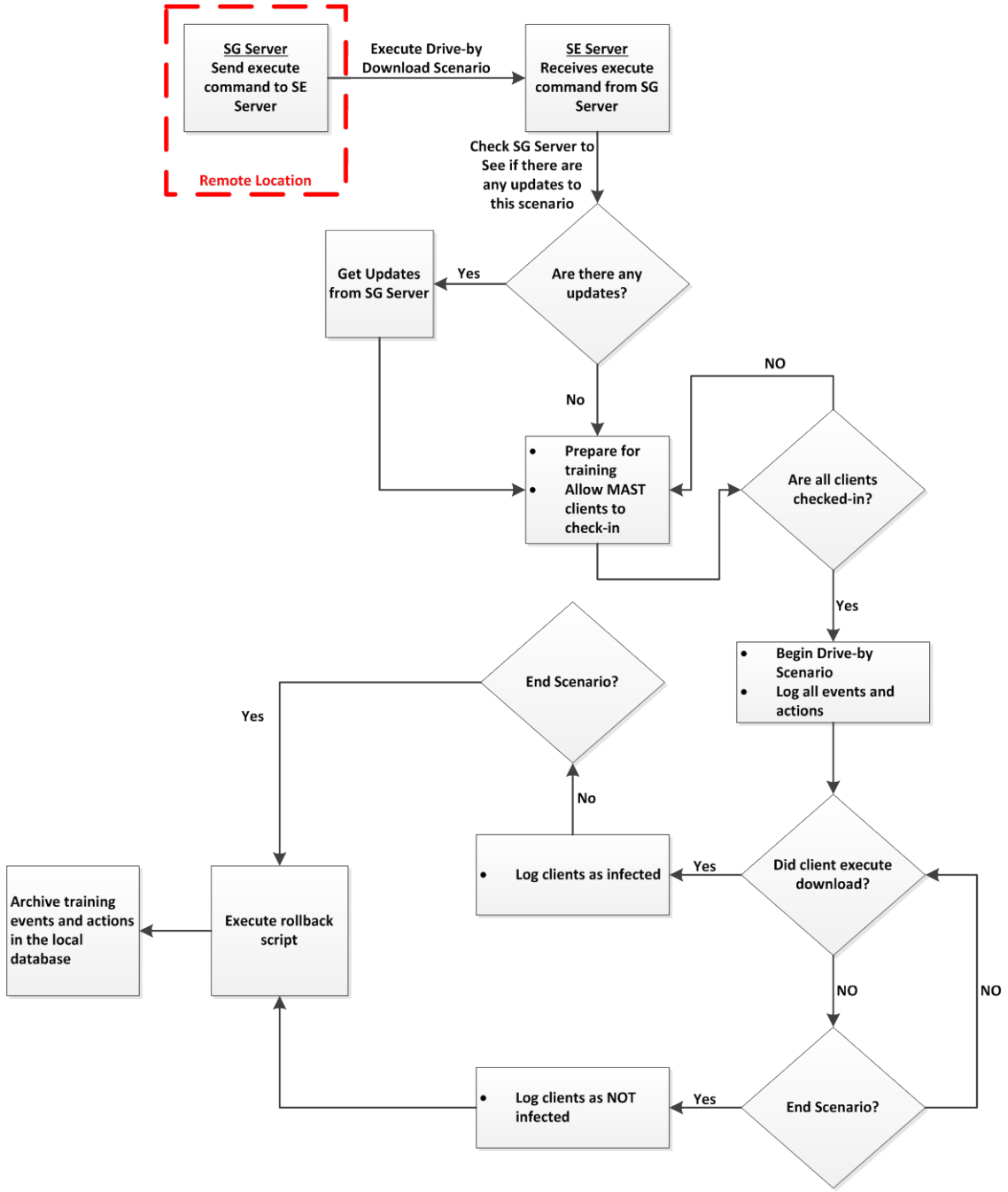


Figure 4. Example of a MAST Simware Scenario. From [4].

2. A Training Scenario Example

In shaping the design criteria for our training scenario, we focused on the COMPOSE network and its security posture. Behind DoD and DON-provided security services, the COMPOSE networks on U.S. Navy ships may be viewed as internal

networks. Like the private sector, DON internal networks are most susceptible to security threats introduced by their users. Security features and policies are in place to obscure and protect the internal networks from the external threats. One vehicle for allowing an external threat to gain access to a COMPOSE network is e-mail. Spam filters and exchange servers are configured to minimize the delivery of e-mails containing malicious attachments or links to COMPOSE inboxes. Even so, unless e-mail is turned off it is impossible to fully eliminate this threat vector. In addition, access to the Internet allows users to utilize web-based e-mail services. In some cases, the security features of web-based e-mail accounts are less protective than those of COMPOSE e-mail accounts. This results in an increase in the vulnerability of the internal network to external threats. Whether opened from a DON-protected e-mail account or a web-based e-mail account, the actions of opening a malicious e-mail occur on the local workstation. Any malicious code tied to the e-mail, unless blocked by network security policies and configurations on the local workstation, executes and runs its course internal to the network. As explained in Chapter II, the code may simply manipulate the user's web browsing session or it may start a series of events that creates an open lane of communication between an external threat agent and the internal workstation.

Limiting the scenario to Group 1 and Group 2 malicious behaviors, we produced the following training scenario:

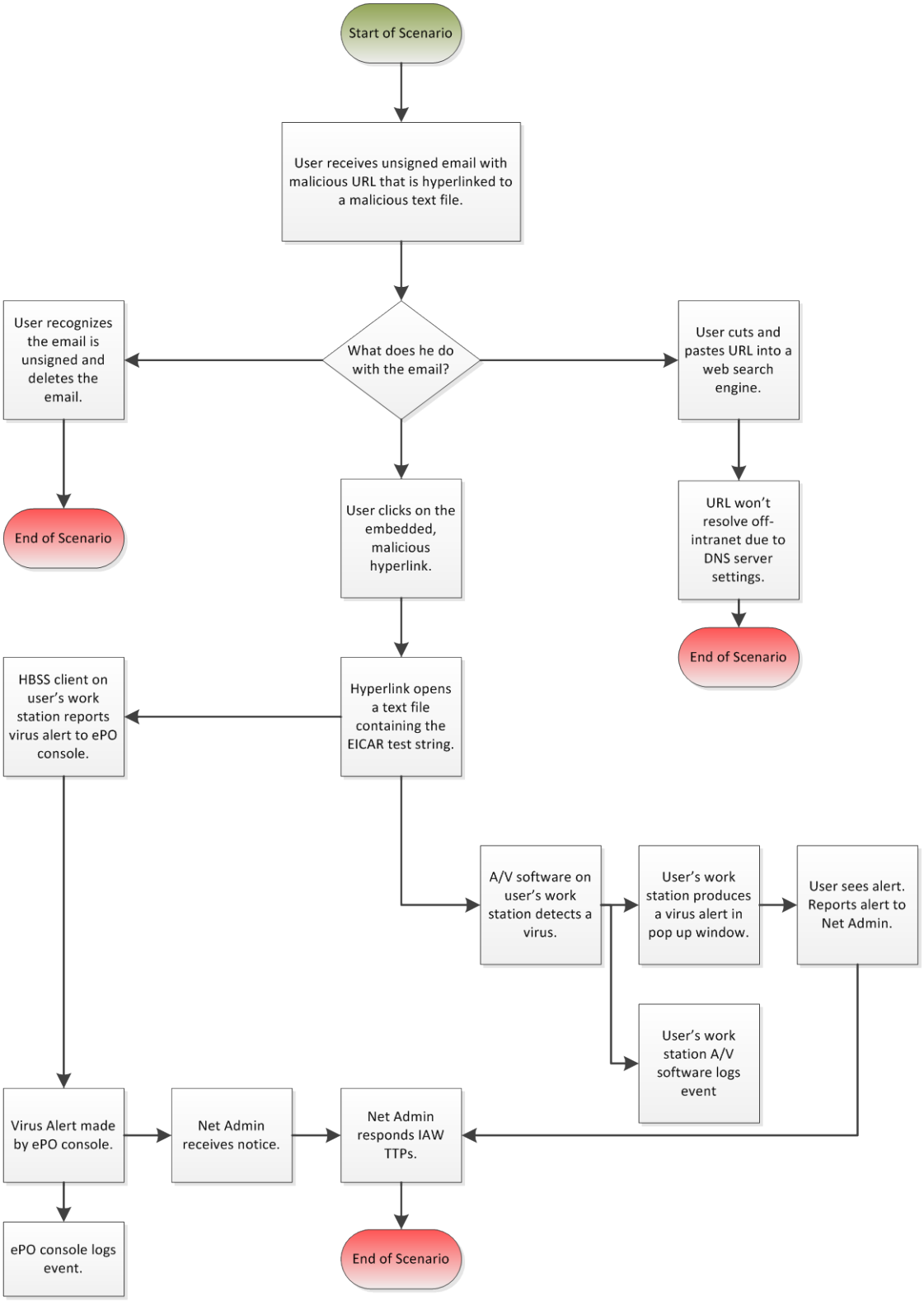


Figure 5. MAST Training Scenario

The training scenario depicted in Figure 5 involves the delivery of an unsigned e-mail. True to the modularity of the MAST design, the e-mail may contain either a malicious link or a malicious attachment. In this scenario, we depict a malicious link Simware module. Per [29], digital signatures are required on e-mails containing active links or attachments. Neither the Simware module containing a malicious link nor the Simware module containing a malicious attachment will produce a signed e-mail intentionally to test whether or not the simulation audience complies with the DoD-mandated Information Systems Security Awareness training. The training prohibits the clicking on hyperlinks or attachments contained in unsigned e-mails because it may install malware on the system.

The scenario begins with the arrival of the malicious e-mail in the user's inbox and may take on several different paths before completion. If the user deletes the e-mail, the training scenario ends. If the user clicks on the malicious hyperlink, the Simware module tied to this training scenario generates a simulated virus. Based on system descriptions, we anticipated a creation of security event alerts by both the host HBSS client and Symantec anti-virus software locally and remotely. The initiation of the simulated virus gives the trainer the opportunity to verify user and network administrator compliance with existing security threat training, tactics, and procedures (TTPs). It also allows for verification of proper configuration and function of the HBSS Suite and Symantec anti-virus software on each participating workstation. If the Simware successfully generates the simulated virus, a properly configured anti-virus application should recognize it, respond to it, and report it both locally and remotely to the Symantec anti-virus server loaded on MASTUSCDC001. The HBSS client application should also generate an alert locally and remotely to the ePO console loaded on the CND-OSE Server. Any discrepancies would require verification of compliance with configuration and security policies and possible troubleshooting.

C. SAFETY AND SCALABILITY OF MAST SIMULATIONS

Scalability and safety of a simulation is essential to the delivery of any training value by the use of said simulation. Scalability of the MAST system, as mentioned

before, was successfully tested and verified by Longoria [4]. The MAST system is designed to provide training opportunities in an operational network. However, no unit commander will authorize the use of the MAST system on their operational network unless they have faith in the ability to secure any active Simware modules and restore their network to its full operational posture when necessary. Navy ships temporarily reduce their operational readiness in support of combat systems training evolutions with full faith that the information systems used to simulated adversarial actions may be returned to their normal operating modes at any moment. Longoria's thesis describes the MAST safety features that deliver the capability to stop any Simware module and fully restore the network at a moment's notice. He described the client check-in feature that identifies the participating workstations. Only network workstations possessing MAST clients that have checked in will receive anything from the SE Server. If a Simware module must be secured before completion of its scenario, the trainer may use the MAST "Kill Switch" available to both a local trainer through the SE Server and remotely to a remote trainer through the SG Server. His research proved its functionality in a virtual network. We extend his descriptions by addressing a fourth safety feature of the MAST system: all MAST Simware modules are contained and controlled.

Use of a simulation that cannot be properly contained and controlled may damage the training value desired from its use. It may also damage the training environment in which it is deployed. It is easier to make this point when considering a training munition used to simulate an actual munition. Even though the training munition contains a fraction of the explosive material found in the larger munition it aims to simulate, if improperly detonated, the force could still result in training audience injury or training space damage. This is not the case with MAST Simware modules. In addition to the capability to secure a simulation and restore the operational network to full capacity when necessary, the safety of the network is baked into each Simware module. For example, existing Simware modules simulating a virus utilize the EICAR anti-malware test file. Created by the European Institute for Computer Anti-virus Research (EICAR), the test file is used to verify the proper operation of anti-virus software in operational networks [30]. The use of the EICAR test file presents no threat to a real network because

it is not a virus. It is a simple DoS program that tells the anti-virus program it is a test file. The code for the file is as follows:

```
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-
ANTIVIRUS-TEST-FILE!$H+H*
```

In the code, you can see the text "EICAR-STANDARD-ANTIVIRUS-TEST-FILE!" This is what is displayed in any alert generated by an anti-virus program when the file is detected. This is just an example of the overall effort to safely implement the MAST system.

D. SUMMARY

This chapter described the design phase of our research. We explained our reason for testing MAST in two separate testing environments: the MAST Test Bed and NCOR. We explained the design of the MAST Test Bed and the research we conducted to ensure it modeled a shipboard network. We then explained the NCOR test environment and its role in supporting the DoN cyber security initiatives. We identified differences between the MAST Test Bed and NCOR and explained how we accounted for these differences in our research. Finally, we explained the design process for the Simware modules used in our test and evaluation of MAST.

In the next chapter, we will discuss the methodology of our test and evaluation of the functionality of the MAST system on the NPS Test Bed and NCOR CVN-65 COMPOSE LANs.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. METHODOLOGY AND RESULTS

This chapter describes our assessment of MAST's compatibility with the COMPOSE environment. The purpose of our research is to determine if MAST is capable of executing the defined simulation in compatibility with the COMPOSE operating system running on a physical network. We begin with a description of the investigative efforts made to better understand the COMPOSE environment. We explain the measures taken to ensure the MAST Test Bed replicates the COMPOSE environment employed by most U.S. Navy shipboard operational networks. We discuss the creation of the MAST Test Bed and the development of the MAST Simware modules. We describe the logical and physical differences between the NCOR and the MAST Test Bed. Finally, for both test environments, we provide the methodology used in our assessments of MAST as well as an analysis of the results from the MAST evaluations conducted in test environment.

A. ESTABLISHING THE TEST ENVIRONMENT

Chapter III outlines the design considerations for the test environments. Having knowledge of what components are used to form the COMPOSE environment is insufficient to create a working implementation of the COMPOSE environment. Installation and configuration details are also needed. For this, we reached out to the Information Assurance and Cyber Security Program Office (PMW-130) and the Tactical Networks Program Office (PMW-160). In addition to generously providing COMPOSE and CND-OSE installation packages with temporary test licenses, both management offices provided technical assistance during the installation and configuration of the MAST Test Bed. Through direct access to SPAWAR installation engineers, we were able to troubleshoot through several configuration issues affecting the exchange server and the HBSS client deployment. Prior to receiving the installation packages, we gained familiarization of the Microsoft® Exchange Server installation and configuration process in one of our classes, CY4700, Cyber Wargame: Blue Force Operations. This class required the students to design and develop the network it would later defend against

attacks from the red team class, CY4710, Cyber Wargame: Red Operations. Given the option of which mail server to utilize in the blue team network, we selected and installed Microsoft®'s Exchange Server 2008. Lessons learned from the deployment of the blue team mail server helped considerably during the COMPOSE installation and enabled us to properly identify installation issues when requesting troubleshooting assistance from the SPAWAR installation engineer.

To facilitate realistic Simware module development, we required a better understanding of HBSS and how it interacts with the COMPOSE environment. To begin, we completed the HBSS introductory training and management courses offered by the Defense Information Systems Agency (DISA) through the CERT® Program's Virtual Training Environment (VTE). To help us with the installation process, the SPAWAR Systems Center, Pacific (SSC Pacific) Training Development and Support Center (TDSC) provided us with installation, configuration and troubleshooting guides for the CND-OSE environment, version 1.2. PMW-130-provided CND-OSE installation engineers helped us resolve issues encountered during our CND-OSE server and HBSS installation, to include the prevention of a potential source of recurring performance issues. The DISA Information Assurance Support Environment (IASE) website supports HBSS versions up to, and including 4.6. The version of HBSS used by shipboard networks is version 4.2 [31]. Had we installed and rolled out policies available for the latest version of HBSS, the environment would not have mirrored those found on U.S. Navy ships, thus resulting in a less authentic experiment. Also, given the unique characteristics of the COMPOSE environment, running a more recent version than HBSS 4.2 may have resulted in errors during testing that were not related to the MAST system or its Simware modules. Finally, just as we were able to benefit from lessons learned in CY4700 during the installation and configuration of our COMPOSE Exchange server, the installation and configuration of the MAST Test Bed HBSS suite was aided by our experiences gained by installing and configuring the blue team network's HBSS suite in CY4700.

Possessing a working replica of a shipboard network in our lab at NPS enabled us to test and evaluate the functionality of MAST and its Simware modules against a

COMPOSE environment. To ensure that MAST and Simware modules were compatible with the NCOR CVN-65 domain, we convened two planning meetings with the staff at NIOC Norfolk. The first was at NIOC Norfolk, the second via video teleconference (VTC). NCOR engineers and red team instructors answered our compatibility questions, explained configuration settings, and even provided constructive feedback on a few of our ideas regarding Simware modules [32].

B. THE MAST TEST BED (NAVAL POSTGRADUATE SCHOOL)

1. MAST Test Bed Construction

As mentioned in Chapter III, the MAST Test Bed was constructed utilizing servers, routers, switches, racks, and workstations readily available in the Computer Science department's reserves. We faced several obstacles while constructing the MAST Test Bed. First, the COMPOSE and CND-OSE software loads required upgrades once installed. Also, the Symantec Anti-Virus ™ software and HBSS ePO required updates to signatures and policies.

One of the safety features of the MAST Test Bed is its lack of connectivity to an outside network. In the interest of time, we decided that the MAST Test Bed would be connected temporarily to the Internet via the NPS Research Network to facilitate downloading necessary patches and signature updates. In order to connect our router to the NPS Research Network, we had to request permission from the NPS Information Technology and Communications Services (ITACS). The request entailed several steps. First, the router's Media Access Control (MAC) address was added to port security policy of the local NPS Research Network switch. Second, ITACS assigned a static IP address to the MAST Test Bed router. The static IP address served two purposes. This allowed the router to be authorized, thus preventing it from being denied access to the NPS Research Network by the anti-router policy. The IP address was also used to identify the MAST Test Bed as exempt from the network access scans conducted by SafeConnect, a proprietary network access control solution produced by Impulse Point [33]. It enables NPS to enforce network access policy compliance by scanning all computing devices requesting connectivity to the NPS Research Network. Any device not

properly configured, running software that is not properly patched, or that is using outdated anti-virus signatures is denied access to the network. Once the MAST Test Bed software was properly patched and anti-virus signatures were updated, the router was disconnected from the NPS Research Network. The static IP address continued to remain authorized and available for reconnection to the NPS Research Network.

2. MAST Simware Module Development

For our testing, we chose to simulate three malicious activities:

- Malicious Port Scan
- Malicious E-mail
- Malicious Pop-Up Window

They provide a simulation that tests MAST's ability to trigger system responses that warrant action, either by the COMPOSE environment in the form of an alert or fail safe response, or by the network administrator or user. A MAST software engineer, Mr. Greg Belli, provided us with a virus Simware module that produces the EICAR test file. A Malicious Pop-Up Window Simware module already existed as part of his ongoing research associated with MAST. He also developed several port scans with different functionalities [7]. The root functionality of all MAST Port Scan Simware modules is provided by the Network Mapper or NMAP utility.

NMAP is an open source application that enables network administrators and security personnel to conduct network surveying and auditing. According to the SANS Institute's white paper, "Conducting a Penetration Test on an Organization," NMAP port scans produce sensitive information about the network including: which operating systems are running on the associated servers and workstations and what services are available on the network based upon which packets do and do not penetrate the network's firewall [34]. The collection of this information from a network is possible because of the exploitation of RFC 793, as explained in Chapter II. In keeping with the constraint of producing simulation for only Group 1 and Group 2 malicious behaviors, none of the Port Scan Simware modules conducted scans that could be considered covert or so slow as to

evade detection. Details regarding Simware module code and its development will be addressed in Mr. Belli's future thesis.

Initial testing of the Port Scan Simware modules during development produced unexpected errors. Investigation identified a conflict in the version of Java MAST was compiled in and the version of Java loaded in the MAST Test Bed COMPOSE environment [35]. MAST originally was compiled using Java version 1.5. The MAST Test Bed COMPOSE environment utilizes Java version 1.6. Once re-compiled with Java version 1.6, all Port Scan Simware modules functioned properly. Additionally, there were anomalies during pre-test development of the Port Scan Simware modules. Initially, the SE server was loaded to the MAST Test Bed File Server (MASTSMS001). When running the Fast Port Scan Simware module, we noticed that, in addition to detecting the scan, the Host Intrusion Prevention System (HIPS) application on the targeted workstation blocked all traffic from the source of the scan (MASTSMS001) for fifteen minutes. This fifteen minute block was part of the HBSS security policies. As a result of the block we were unable to run an additional Simware module for fifteen minutes.

3. Test Methodology

Our test methodology is influenced by the Operational Test Director's Manual (COMOPTEVFORINST 3980.2) [9] and draws from the quantitative testing process presented by Hayes [6]. As stated in [6], there were three overall objectives of our quantitative testing of the MAST Software. The first objective was to assess the compatibility of the MAST Software with the COMPOSE environment while idle. The second objective assessed MAST's compatibility with the COMPOSE operating system while executing Simware modules. The third objective was to verify the functionality of the MAST Kill Switch on a physical network. As stated in Chapter III, Longoria's research proved the functionality of the MAST Kill Switch in a virtual environment [4]. One of the core purposes of testing MAST on a physical network is to verify that the commands tied to implementation of the Kill Switch were carried out properly. Operation of the Kill Switch at any point during the execution of a Simware module should result in

immediate halt of the module and a full restoration of the network to its operational status prior to commencement of the Simware module.

In shaping the test plan for the MAST Test Bed portion of our research, we divided our test plan into three phases: installation of the MAST Software, execution of Simware modules, de-installation of the MAST Software. During each phase, our test plan required collection of information that established and documented the operational state of the COMPOSE environment. This operational state was treated as the baseline status of the test network. Knowledge of this baseline status would then enable us to determine whether or not the MAST Software caused any adverse impact to the network at any point while following the test plan. This might indicate an incompatibility between the MAST Software and the COMPOSE environment. The baseline information also enabled the accomplishment of the first objective of our quantitative testing.

The scope of the NPS Test Bed test plan is the NPS Test Bed, the MAST SE server, the MAST clients, and the associated malicious software simulations or Simware modules. Critical operational issues to be examined are:

- Ensure MAST is compatible with a COMPOSE environment in an idle state
- Ensure execution of the MAST Software while running Simware modules does not hamper functionality of a COMPOSE environment
- Ensure the ability of the MAST Kill Switch to fully restore the test environment to its previous operational status.

The following sections provide detailed descriptions of the three phases of our testing.

4. MAST Installation Test Procedures

a. Pre-Deployment of MAST

The objective of this portion of the test procedure is to ensure full knowledge of the COMPOSE environment running on the test network. For the purpose of these test procedures, the phrase “participating workstations” refers to all testing network’s workstations that will be monitored during the test (Work Stations 01 – 06, See Figure 6). Also, for the duration of the Simware module testing, “testing workstations”

refers to the group of network workstations running the MAST client (Work Stations 01 – 04, See Figure 6).

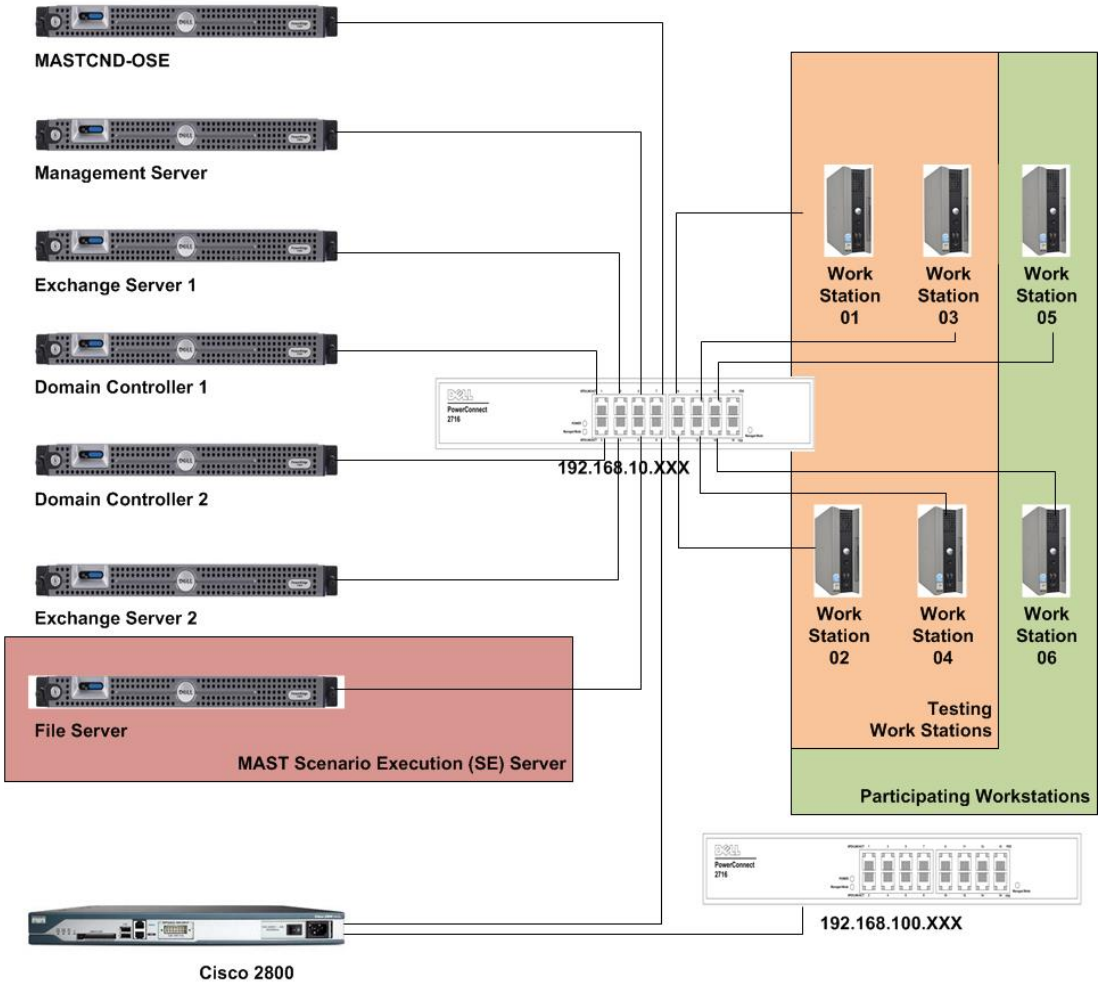


Figure 6. NPS MAST Test Bed Network Topology with Device Role Identified

In the case of the NPS Test Bed, there are six physical workstations. Four workstations will be MAST workstations that will run the MAST clients and consist of the audience of workstations in the simulation modules. Two workstations will act as controls. They will not be loaded with the MAST client. They will not participate in the simulation modules. All simulation activity should be undetected on the control machines. All six workstations are clarified as “participating workstations.”

Step	Procedure
1	Identify server loads on domain controllers, servers, and workstations
2	Map network topology
3	Inspect event viewer on all servers and participating workstations
4	Identify critical services on Domain Controllers 1 and 2
5	Identify critical services on Exchange Servers 1 and 2, Management Server, and File Server
7	Verify file, registry setting, etc. (See Registry Integrity Checks)
8	Verify functionality of software applications including Microsoft Suite (Word, Excel, Access, PowerPoint, Outlook, and IE) (See MS Suite Functionality Checks)
7	Conduct Host Resource Usage Checks (See Host Resource Usage Checks)
10	Conduct additional TBD inspections if defects or disruptions are discovered while accomplishing Test 2.1 steps 1 - 9

Table 2. MAST Pre-Installation Test Procedures

b. Post-Deployment of MAST

This portion of the installation test procedure assesses the impact to the NPS Test Bed, the MAST workstation, and the participating workstations loaded with MAST clients.

Step	Procedure
1	Start SE server on the MAST Server.
2	Start MAST clients on participating workstations
3	Inspect event viewer on all servers, and all participating workstations
4	Identify critical services on Domain Controllers 1 and 2
5	Identify critical services on Exchange Servers 1 and 2, Management Server, and File Server
6	Verify critical services on all participating workstations
7	Verify file, registry setting, etc. (See Registry Integrity Checks)
8	Verify functionality of software applications including Microsoft Suite (Word, Excel, Access, PowerPoint, Outlook, and IE) (See MS Suite Functionality Checks)
9	Conduct Host Resource Usage Checks (See Host Resource Usage Checks)
10	Conduct additional TBD inspections if defects or disruptions are discovered while accomplishing Test steps 1 - 9

Table 3. MAST Post-Installation Test Procedures

(1) Microsoft® Suite Functionality Checks. These functionality checks are conducted on the MAST workstation and all participating workstations onto which the MAST client was loaded. For each machine, the functionality checks are conducted for each MS Suite Program (e.g., Word, Excel, PowerPoint, etc.) loaded on that machine.

Step	Procedure
1	Open Event Viewer. Monitor Applications Log throughout functionality checks
2	Open MS Suite program
3	Create a test file using this MS Suite program
4	Save the file locally
5	Save the file to the network share drive
6	Close MS Suite program
7	Open test file from its local storage location
8	Open test file from its network share drive location
9	(For MS Exchange) Send e-mail to test administrator account
10	(For MS Exchange) Send e-mail from test administrator account to client account
11	Conduct additional TBD inspections if defects or disruptions are discovered while accomplishing Test 2.2.1 steps 1–10

Table 4. MS Suite Functionality Checks

(2) Host Resource Usage Checks. These resource usage checks are conducted on the MAST workstation and all participating workstations onto which the MAST client was loaded.

Step	Procedure
1	Identify the amount of disk space consumed by the application on MAST server and workstations.
2	Inspect the % of CPU usage on MAST server and workstations
3	Inspect the pages/sec memory usage on MAST server and workstations
4	Inspect send/receive bytes count of network adapter on MAST servers and workstations
5	Verify no HIPS warnings occurred during MAST software installation on MAST sever and workstations
6	Conduct additional TBD inspections if defects or disruptions are discovered or significant increase in CPU usage occurs while accomplishing Test steps 1–6

Table 5. Host Resource Usage Checks

(3) Registry Integrity Checks. The following integrity check is conducted on any server or workstation onto which the MAST client and server is loaded. For each machine, the registry checks are conducted for each HKEY_LOCAL_MACHINE registry subtree on that machine.

Step	Procedure
1	Start Regedt32.exe
2	Select each registry root HKEY_LOCAL_MACHINE
3	From the menu bar, File and then Export
4	Click Save as type
5	Select Text files
6	Save the file as reg.old or, for post MAST install, reg.new
7	Run Windiff from the Microsoft Support\Tools and compare the two files, reg.old and reg.new. (omit for preinstall)

Table 6. Registry Integrity Checks

5. Simware Module Test Procedures

The testing workstations were monitored for data involved in the simulation activity. The term “participating workstations” referred to the testing workstations and the control workstations. Data was collected from all participating workstations during each Simware module test procedure. All Simware modules were tested using the following test procedure:

Step	Procedure
1	Capture system time
2	Start SE server
3	Start MAST client on testing workstations
4	Select and execute Simware module from SE Server
5	Verify start of nmap on MAST workstation
6	Collect data from servers and participating workstations. Save to “Test Data” folder
7	Inspect event viewer on servers, and all participating workstations
8	Identify critical services on Domain Controllers 1 and 2
9	Identify critical services on Exchange Servers 1 and 2, Management Server, and File Server
10	Verify critical services on MAST workstation and all participating workstations
11	Verify file, registry setting, etc. (See Registry Integrity Checks)
12	Verify functionality of software applications including Microsoft Suite (Word, Excel, Access, PowerPoint, Outlook, and IE) (See MS Suite Functionality Checks)
13	Conduct Host Resource Usage Checks (See Host Resource Usage Checks)
14	Conduct additional TBD inspections if defects or disruptions are discovered while accomplishing Test steps 1 - 13

Table 7. Simware Module Test Procedures

During our Simware module testing we ran simulations that captured the three malicious activities mentioned above:

- Malicious Port Scan
- Malicious E-mail
- Malicious Pop-Up Window

Explanation of the expected functionality of the individual Simware Modules used in our testing is provided next. In summary, tested the following Simware modules:

Malicious Activity	Simware Module	Summary
Malicious Port Scan	Services Only Scan	Only scans for ports listed in NMAP's services file
	Fast Port Scan	NMAP scans full range of ports in shortest time
	Covert PING Scan	Sends SYN and ACK flags in place of TCP Ping
	XMAS Tree Scan	Sends FIN, URG, and PSH packets to identify open ports
Malicious E-mail	Malicious E-mail with Text File	Generates e-mail with text file attachment containing EICAR test string.
	Malicious E-mail with Batch File	Generates e-mail with batch file, which produces the EICAR test string when executed.
Malicious Pop-Up Window		Clicking on window's buttons results in EICAR test string generation on host workstation.

Table 8. Simware Module Summary Table

We assume that a more detailed explanation of the software that delivers this simulation will be provided in Belli's forthcoming thesis.

a. Port Scan Simware Modules

The TCP port scan executed by the port scan modules should be detected and blocked by the Host Intrusion Protection System (HIPS) on each testing workstation. The detection and blocking action should be reported locally on each workstation and also at the ePO management console. MAST SE server should receive reports from each MAST client indicating time scan started and time scan stopped.

(1) Services Only Scan Simware Module. This module conducts a TCP port scan using NMAP and the switch "-F." It limits the port scan on a named IP address to only those ports listed in the NMAP services file.

(2) Fast Port Scan Simware Module. From his tutorial on using NMAP to conduct stealthy port scans, Bennieston explains how the timing with which NMAP conducts its port scans may be controlled using the "-T" parameter. "There are six predefined timing policies which can be specified by name or number (staring with 0, corresponding with Paranoid timing). The timings are Paranoid, Sneaky, Polite, Normal, Aggressive and Insane. A –T Paranoid (or –T0) scan will wait (generally) at least 5 minutes between each packet sent" [36]. For this module, we used the parameter of –T5, which equals to the timing setting of Insane. It covers all ports very quickly, possibly at the expense of losing some data.

(3) Covert PING Scan Simware Module. This scan utilizes the PING switch (-P) to send SYN and ACK packets in the place of a TCP Ping. This module will execute the "NMAP –PS" followed by the "NMAP –PA," both across the same range of IP addresses. This scan simulates an attacker attempting to establish if a host is on the network. The result is similar to using a PING, without sending a PING packet. This captures the intent of the PING utility on a network where security policies might prevent PINGs between workstations.

(4) XMAS Tree Scan Simware Module . This scan shapes the TCP frame sent from the SE server to contain the flags FIN, URG, and PUSH. Per RFC 793, the incoming TCP segments, not containing RST flags, should result in a responding TCP segment containing a RST flag for any port CLOSED on the workstation [23]. If the port is OPEN or if the workstation contains a stateful firewall, the NMAP report for this scan lists the port as "OPEN | FILTERED" [23]. It is called a XMAS Tree scan because the TCP segment flags that are on (00101001) resemble the blinking lights of a Christmas tree.

b. Malicious E-mail Simware Modules

The detection and response to malicious e-mails generated during these simulations depends on which Simware module is executed. Both begin with the SE server directing a MAST client to generate the malicious e-mail. The malicious e-mail is the vehicle for a malicious attachment. Both should result in the generation of the EICAR test string (see Chapter III for explanation).

(1) Malicious E-mail with Text File Simware Module . After producing the malicious e-mail, the MAST client should generate a text file. The text file should contain the EICAR test string. Who the MAST client sends this malicious e-mail with EICAR text file to is determined by the e-mail addresses entered in the module prior to execution. Once the e-mail is transmitted, the automated scan conducted by the anti-virus software loaded to the MAST client's host workstation should identify the EICAR test string as a virus and strip the text file. This tests the proper configuration and operation of the Symantec anti-virus client application. Should the text file not be stripped and reach the intended recipient mail boxes, detection and quarantine of the EICAR test string should occur locally if the user selects to read the text file attached to an unsigned e-mail (in violation of DoD IA policy).

(2) Malicious E-mail with Batch File Simware Module. In this scenario, the designated MAST client creates two variables each containing half of the EICAR test string. A batch file is then created that combines the two variables to form the EICAR test string once executed. Given the prior knowledge that a batch file would most likely be stripped by the host workstation's Symantec anti-virus application during scanning, some social engineering is employed to enable the attachment to bypass the automated scans. The MAST client saves the batch file (.bat) as a text file (.txt) attachment to the e-mail. In the message body, the e-mail contains instructions to convert the .txt file to a .bat file in order to view the contents. Should the user do so, the .bat file should execute producing the EICAR test string. Local detection and quarantine of the file should occur as previously discussed.

c. Malicious Pop-Up Window Simware Module

This module is designed to test the end user's compliance with the security practices outlined in the DoD-mandated Information Systems Security Awareness training. The MAST client loaded to the testing workstations should locally produce a pop-up window. These windows should appear once the SE server executes the scenario and without any interaction from the user on the testing workstations. We simulate users at two of the testing workstations clicking on the button contained in the pop-up window to test the module's ability to simulate a resultant infection of the host machine. Clicking

on the radio button should result in local generation of the EICAR test string. If the EICAR test string is generated, local detection and quarantine of the file should occur as previously discussed.

d. MAST Kill Switch Testing

Once all testing of the Simware modules are complete, each Simware module will be run again, this time testing the functionality of the MAST Kill Switch. The Simware Module Test Procedures (Table 7) will act as the MAST Kill Switch Test Procedures with one revision. A new step, "Utilize the MAST Kill Switch from SE server," will be added following step 11 of Table 7. The remaining portion of Table 7 will be completed as part of this test.

e. MAST De-installation Procedures

MAST is a portable application that does not actually install any software. As a portable application, MAST does not write any configuration files to the host operating system and it does not write any data to the system registry. We copied MAST from a portable media (CD) to the root, "C," drive of the host using the copy-and-paste functions in the windows environment. Since there is no installation of program files associated with MAST, there is no de-installation procedure for MAST. When verifying MAST's compatibility with the COMPOSE environment in an idle state, we had to start the MAST server and all participating workstations.

6. Test Results

Overall, our testing verified the capability of MAST while functioning on a physical network utilizing the COMPOSE operating system. Installation of MAST on the MAST Test Bed physical network did not adversely impact the COMPOSE operating system. During the execution of the Simware modules, neither the dynamic state of the SE server nor the MAST clients produced any indication of negative effect on the MAST Test Bed servers or workstations that would impinge upon their operational status. Additionally, when utilized, the MAST Kill Switch successfully cancelled the running

Simware module and returned the MAST Test Bed to the operational status it was in prior to commencement of the Simware module.

a. MAST Functionality

As anticipated, the MAST installation went smoothly and was uneventful. Given the success of past theses whose research rested on the firm foundation of a properly installed and functioning MAST, there was little reason to expect significant compatibility issues. Potential sources of incompatibility were errors due to variance in COMPOSE versions and errors due to configuration errors. Lieutenant Commander Neff and Captain Longoria conducted their research on a virtual implementation of the COMPOSE version 3.0 environment [3], [4]. The MAST Test Bed uses COMPOSE version 3.5. As mentioned earlier, a compatibility issue involving the JAVA version used to compile the Simware Module was identified and remedied during pre-test Simware module development. In addition to the variance in the COMPOSE versions, ensuring a proper installation and configuration of the COMPOSE version 3.5 on the MAST Test Bed was vital in creating a controlled testing environment. The early involvement of SPAWAR installation engineers, our Fleet expertise and familiarity with the COMPOSE environment, and a slow, methodical, and well-managed installation process all contributed to the successful creation of a proper replica of an operational shipboard network. Our functionality checks produced no identified defects or disruptions to the availability and operation of the MS Suite. The host resource usage check verified the MAST installation presented no significant change in the CPU usage of either the network servers or workstations. With the exception of the initial jump in CPU usage associated with starting the Windows Task Manager application, the CPU usage throughout the installation process averaged 2%.

b. Simware Module and MAST Kill Switch Execution

The successful installation of MAST verified its compatibility with a COMPOSE environment while in an idle state. The successful execution of all tested Simware modules not only verified MAST's compatibility with a COMPOSE environment while operational but also verified its ability to provide a realistic simulation

of malicious activity in a non-virtual network. The compatibility and functionality of the Simware modules was confirmed as a side effect of the Mast Simware module development prior to testing. However, testing of each Simware module in accordance with the test procedures prescribed in [6] was necessary to validate the findings of our research.

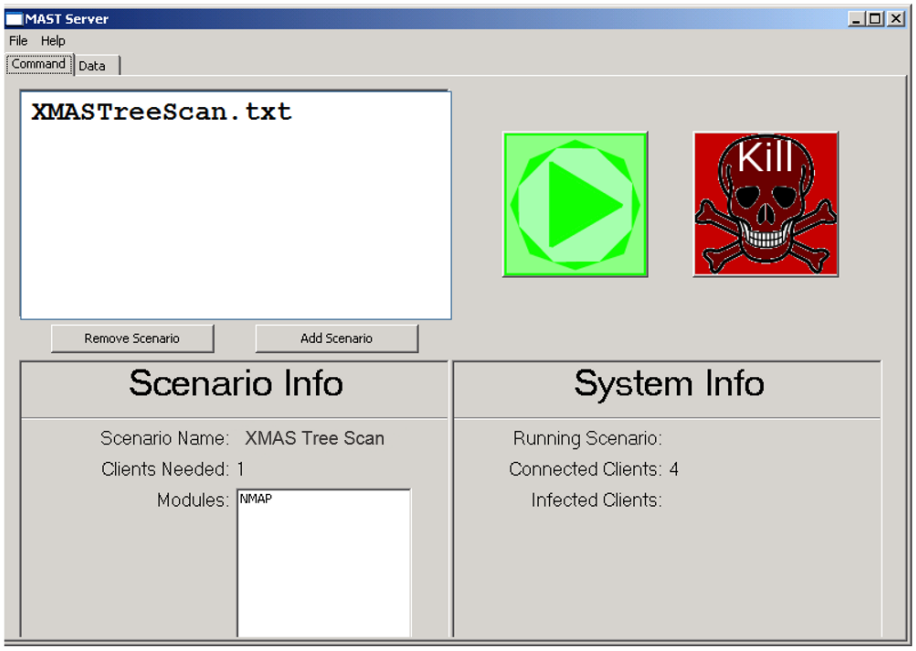


Figure 7. MAST Graphical User Interface (GUI)

Each of the Simware modules identified in Section 5, Simware Module Test Procedures, were tested twice: once to verify the Simware's ability to properly simulate the intended malicious activity, and, second, to verify the proper operation of the MAST Kill Switch. Figure 7 shows the MAST GUI prior to the execution of one the XMAS Tree Scan Simware module tests. The GUI indicates the scenario to be executed is the XMAS Tree Scan. Four MAST clients are connected but only one was needed to run the scenario. Once the green "play" button was clicked, it was replaced on the GUI by a red "stop" button, and the scenario initiated a XMAS Tree Scan as explained

previously. Like all Simware modules, CPU usage monitoring presented no indication that execution of the XMAS Tree Scan Simware module required significant use of host resources.

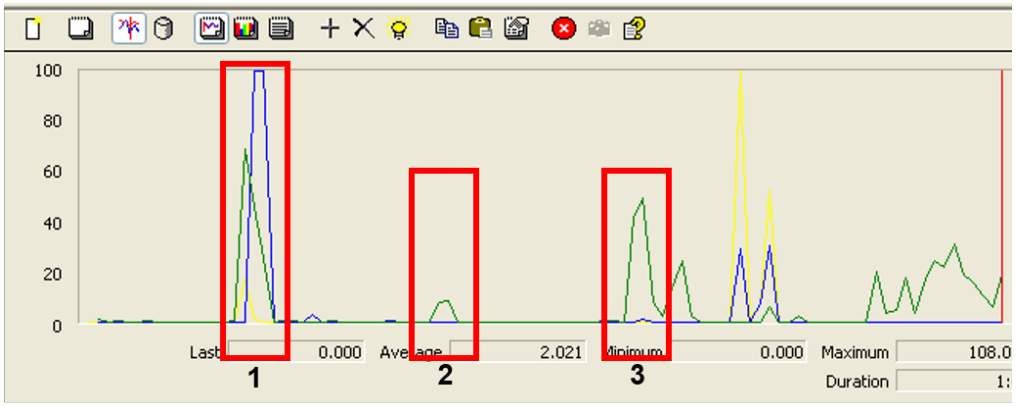


Figure 8. CPU Utilization of Workstation Conducting XMAS Tree Scan

Figure 8 shows the CPU performance graph for the SE server host. Box 1 shows the initial spike in CPU usage associated with starting the SE server. Box 2 shows the MAST "heartbeat," or signal used to communicate information to and from the connected MAST clients. Box 3 shows the beginning of the XMAS Tree Scan using NMAP. CPU usage on the SE server host machine only temporarily increased and aligned with the anticipated CPU usage for this Simware module.

Figure 9. CPU Utilization of Workstation Detecting XMAS Tree Scan

In Figure 9, we see the CPU performance graph from one of the participating workstations running a connected MAST Client. The workstation's IP address was listed in the XMAS Tree Scan Simware module as one of the target IP addresses. Box 1 shows the MAST heartbeat verifying a live connection with the SE server. Box 2 shows the increase in CPU usage associated with the host machine's HIPS application detecting and blocking the inbound port scan. All workstations targeted by the XMAS Tree Scan Simware module showed similar CPU performances and received the same spike in CPU usage during detection of the attack. Figure 10 shows the alert window produced by the HIPS application on the target workstation.

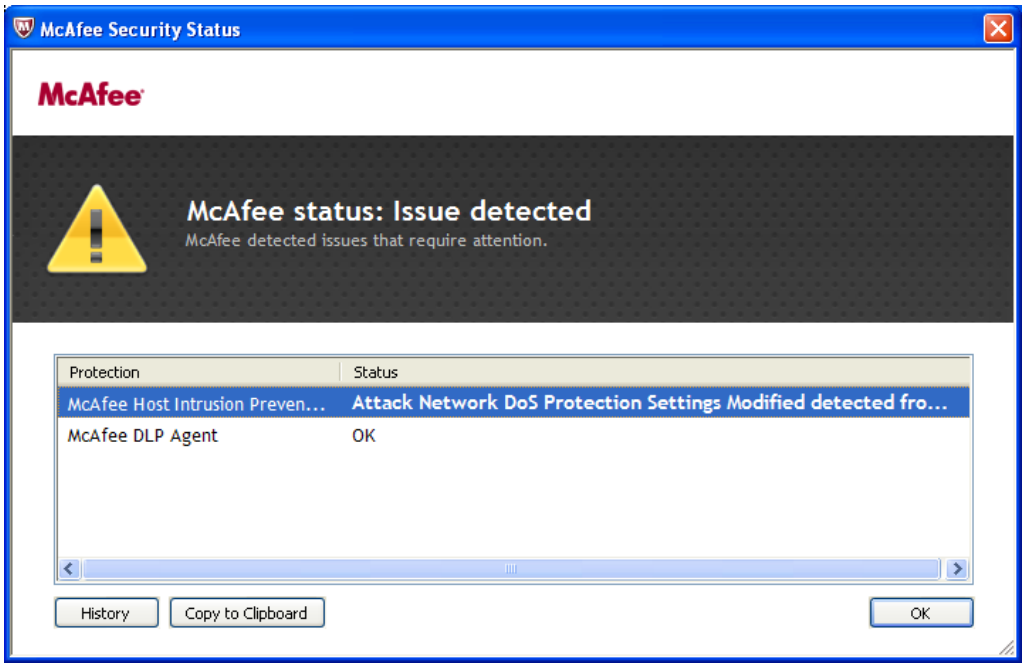


Figure 10. McAfee Notification of Detected Attack

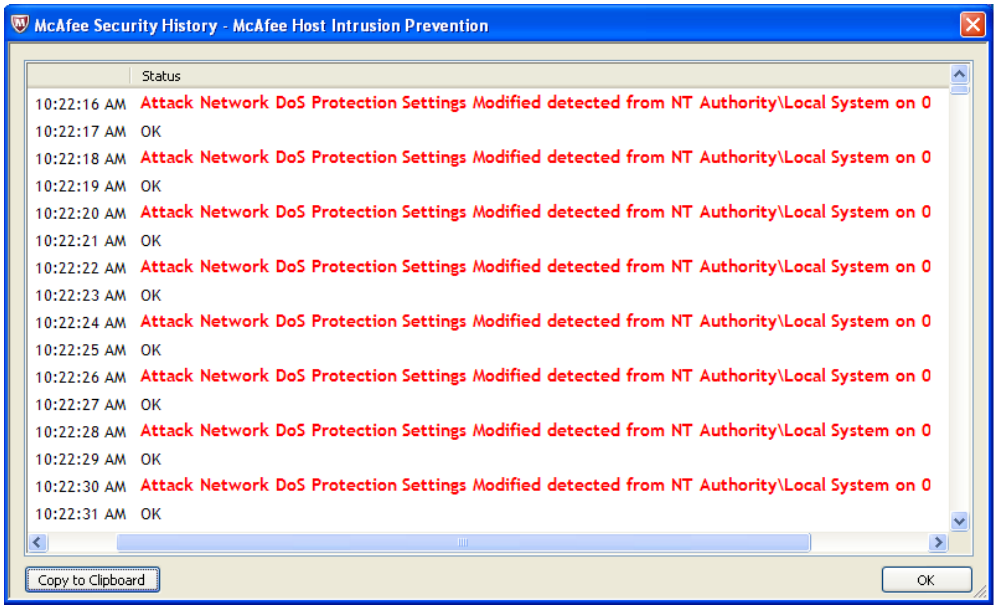


Figure 11. History Log of the Attacked Workstation's HIPS agent

An additional indicator of the scan is found in the history logs of the HBSS-generated McAfee host client loaded to the workstation (discussed in Chapter III). Figure 11 shows the continued reporting of the presence of the scan as well as the

continued verification of the functionality of the McAfee Data Loss Prevention (DLP) Agent. When running a Simware module that could result in the simulated infection of a workstation, like the Malicious Pop-up Window, the MAST client of the simulated infected machine would send an indication of the infection back to the SE server. Infection of the connected clients was successfully reported and tracked during our testing. Figure 12 shows the GUI during the testing of the Malicious Pop-Up Window Simware module (labeled here as the "drivebydownload.txt"). During the test, we clicked on the pop-up window's button. This triggered a simulated infection of the workstation, resulting in the generation of the EICAR test string. The infection was reported by the MAST Client to the SE server and indicated on the GUI.

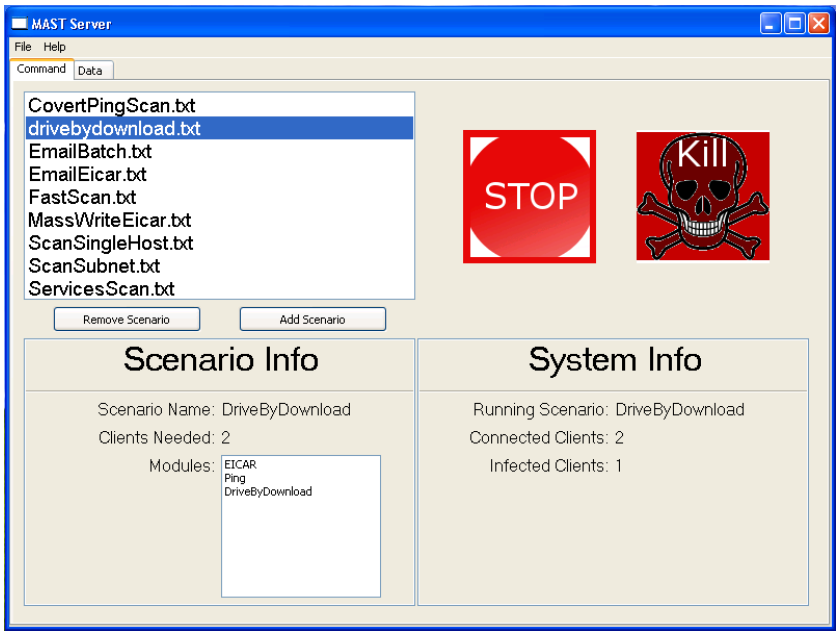


Figure 12. MAST GUI Indicating an Infected Workstation. From [35].

Once the Simware module completed the intended simulation of malicious behavior, the MAST clients remained connected to the SE server. Once the local Safety Observer clicked on the MAST GUI's "stop" button, the roll back sequence commenced. The module was stopped, its termination was logged in the MAST database, and any scripts deployed by the SE server to a MAST client were rolled back. However, the "heartbeat" or connection between the MAST clients and the SE server was maintained.

During the tests of the MAST Kill Switch, we observed the same Simware module roll back sequence. The clear difference between the two functions is the termination of the MAST clients on all participating workstations. Utilizing the "stop" function enables the Safety Observer to load an additional Simware module and continue simulations. Utilizing the MAST Kill Switch sent a termination signal to the participating workstations to close the MAST clients. In each test of the MAST Kill Switch, the connection between the MAST clients and the SE server was severed and the host workstations were fully restored to their previous operational status.

C. NAVY CYBER OPERATIONS RANGE (NCOR) (NIOC NORFOLK, VA)

Testing of MAST on the NCOR was a success. We tested the same three Simware modules: the port scan, the malicious e-mail, and the malicious pop-up windows. Equipped with these Simware modules, we simulated many different forms of malicious activity. For port scans, we utilized the same NMAP flags to simulate Services Only, Fast, and XMAS Tree scans of the NCOR workstations. Like the MAST Test Bed tests, we needed scans that were noisy, meaning they executed port scans that would be easily detected by the McAfee HIPS application. Using the Malicious E-mail Simware, we simulated an unsigned e-mail containing a malicious attachment sent to the victim e-mail account from both internal and external e-mail accounts. The Malicious Pop-Up Window and its ability to simulate viral infection of the host machine was also successfully verified. After minor configuration changes, software updates, and a few Simware adjustments, we successfully ran all tests in accordance with the test plan described by [6]. Functionality of the Kill Switch was tested and verified for all Simware. Attacks were simulated both from one machine to another and from one machine to many.

1. NCOR vs. MAST Test Bed

Testing the Simware on the MAST Test Bed gave us the proper baseline necessary for a successful test at the NCOR. Even so, the successful test and evaluation of our Simware on the MAST Test Bed did not negate the need to test MAST and its Simware on the NCOR. The NCOR is an official DON cyber range. It is certified and accredited to access the Global Information Grid. NCOR has supported several Joint and

National cyber exercises. Testing MAST and its Simware modules on the MAST Test Bed provided clear indication of its compatibility with our instance of a shipboard-simulating COMPOSE environment. Testing of MAST and its Simware on the NCOR gave our results the authenticity that only results from operational testing on a certified and accredited network.

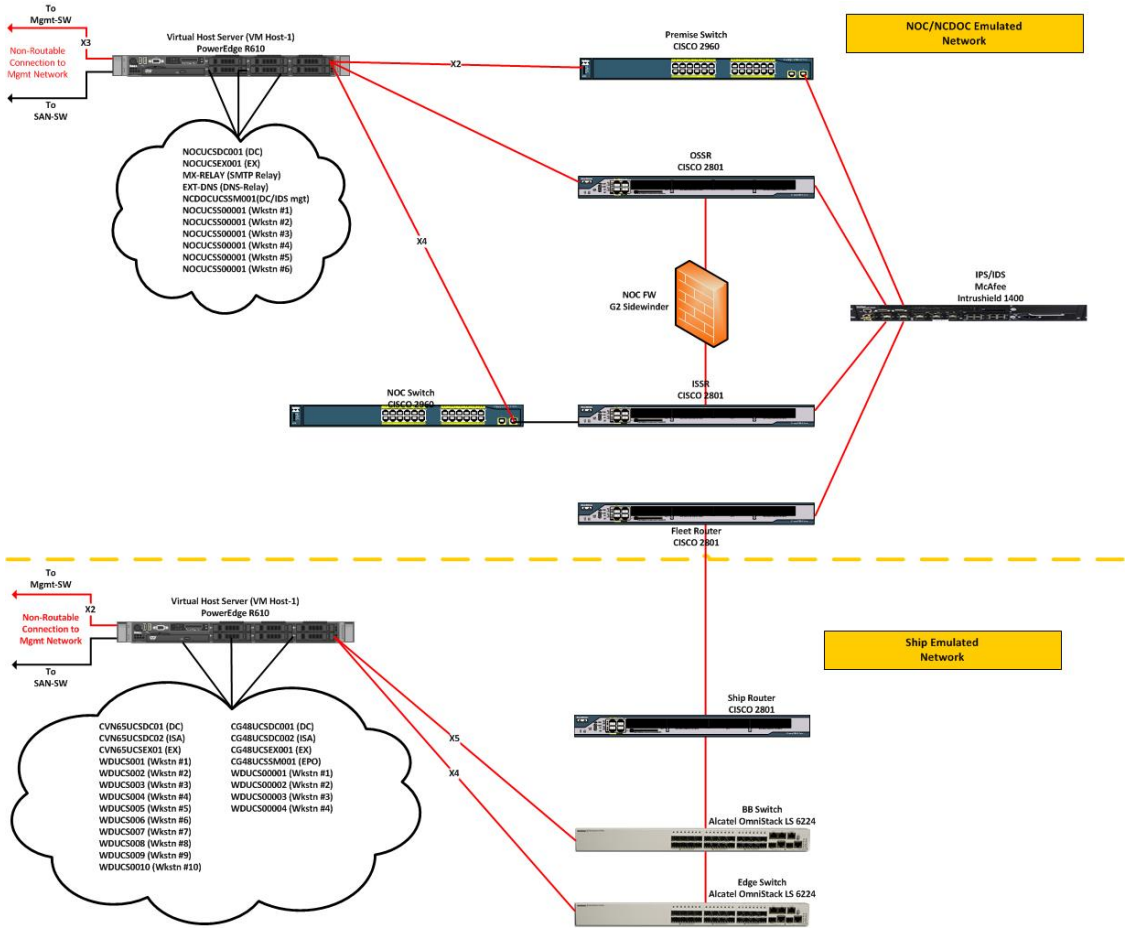


Figure 13. NCOR Network Topology. From [28].

We used different versions of the COMPOSE operating system on NCOR and the MAST Test Bed. NCOR COMPOSE version was 3.0.1. The MAST Test Bed COMPOSE version was 3.5. No errors in MAST functionality or Simware module execution were anticipated given the success of previous MAST research conducted with the same COMPOSE version as NCOR. The MAST Test Bed is a physical network. In

order to deliver the flexibility required of the NCOR, the CVN-65 and CG-48 networks were virtual networks. Figure 13 depicts the NCOR topology. Our testing was conducted on ship-emulated networks housed by the VM Host-1 (See bottom left-hand corner).

2. Test Methodology

The objective of the NCOR testing was to assess MAST's compatibility with COMPOSE and verify the MAST Kill Switch functionality. The same test plan used during testing on the MAST Testbed at NPS was used during NCOR testing; however, an additional Simware module test was added and will be discussed below. The scope of NCOR testing was the NCOR, the MAST SE server, the MAST clients, and the associated Simware modules. This differed from the scope of the MAST Testbed because the NPS Testbed was eliminated and the NCOR was added. The critical operational issues of this test were the same.

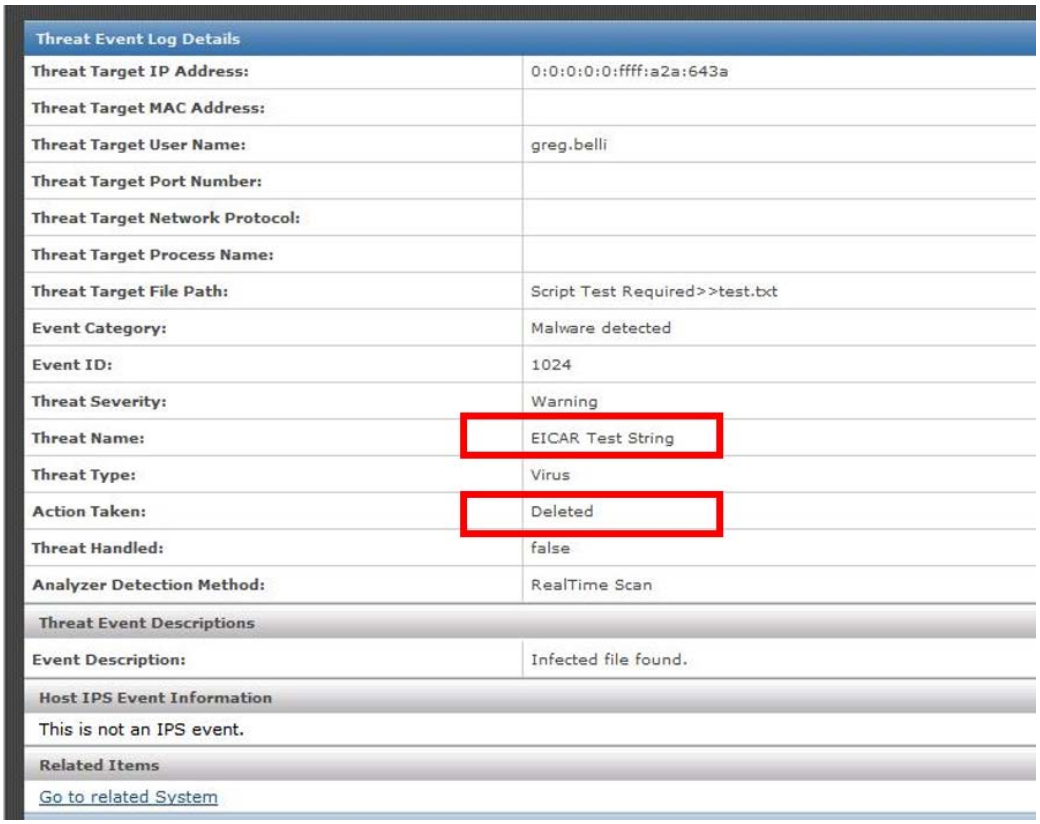
In addition to the Simware module tests conducted during testing on the MAST Test Bed, we added one test to the NCOR testing. We tested the ability to externally generate a malicious e-mail and send it to an internal e-mail account. During our introductory conference with the NCOR engineers, we discussed the feasibility of this test. It was decided that sending an e-mail from an e-mail account external to NCOR (e.g., from an NPS e-mail address) to a NCOR e-mail account was not possible [37]. However, a user account with e-mail address was created on the NCOR's CG-48 domain. Sending an e-mail generated by MAST on the CG-48 domain to an e-mail account on the CVN-65 domain would meet the intent of the test of an external e-mail exchange while remaining internal the NCOR. The MAST Installation Procedure and Simware Module Test Procedures were followed. In this case, host resource usage was conducted on both the CG-48 workstation and all CVN-65 workstations hosting connected MAST clients.

3. Test Results

Our NCOR testing achieved the same level of success as our tests on the MAST Test Bed. MAST's compatibility with the NCOR COMPOSE load was proven and its Kill Switch successfully rolled back all simulations conducted on both the CVN-65 domain and CG-48 domain. Per the training scenario discussed in Chapter III, it was

necessary to prove that MAST Simware modules were able to produce actionable indications to stimulate responses from both users and network administrators. Our NCOR and MAST Test Bed tests verified both local and remote indication of the malicious behaviors simulated by the Simware modules.

The malicious behaviors simulated by MAST Simware modules were reported by the attacked workstation's McAfee host clients to the HBSS ePO management console. In the management console, threat reports were either logged in the Threat Event Log or the IPS Log. Viral infection simulations tied to a Malicious Pop-Up Window or Malicious E-mail Simware module were registered in the HBSS Threat Event Log. All port scans tied to a Port Scan Simware module were registered in the HBSS Host IPS Log. Figure 14 shows the individual Threat Log entry for one of the EICAR Test String detections.



Threat Event Log Details	
Threat Target IP Address:	0:0:0:0:0:ffff:a2a:643a
Threat Target MAC Address:	
Threat Target User Name:	greg.belli
Threat Target Port Number:	
Threat Target Network Protocol:	
Threat Target Process Name:	
Threat Target File Path:	Script Test Required>>test.txt
Event Category:	Malware detected
Event ID:	1024
Threat Severity:	Warning
Threat Name:	EICAR Test String
Threat Type:	Virus
Action Taken:	Deleted
Threat Handled:	false
Analyzer Detection Method:	RealTime Scan
Threat Event Descriptions	
Event Description:	Infected file found.
Host IPS Event Information	
This is not an IPS event.	
Related Items	
Go to related System	

Figure 14. HBSS Event Detail of EICAR Test String Detection

D. SUMMARY

In this chapter, we discussed the methodology applied to the testing of MAST's compatibility with the COMPOSE environment operating on a simulated, shipboard, physical network. We also tested its ability to produce a realistic simulation of malicious behavior and its ability to terminate that simulation and fully restore the network to its previous operational status. The tests conducted on the MAST Test Bed verified MAST's ability to provide network users and administrators with realistic malicious scenarios for training or assessment purposes without detrimental impact to the operational readiness of its host network. Our successful tests on the certified and accredited Navy Cyber Operations Range re-enforced our findings.

In Chapter V, we provide the conclusions drawn from our research. In addition, we provide guidance received during our research on how to improve upon MAST's usefulness to the Fleet. Finally, we will close with lessons learned from our research and suggestions for future work.

V. CONCLUSIONS AND FUTURE WORK

A. SUMMARY

In this thesis, we analyzed the compatibility of MAST with the COMPOSE environment running on a physical network modeled after a shipboard operational network.

In Chapter II, we set the foundation for understanding the threats MAST aims to simulate. We identified the three fundamental objectives of network security. We explained how vulnerabilities act as imperfections in a network's armor exposing it to various threats. The nature of threats present on the Internet today has caused a shift from a binary concern for either viral infection or worm propagation to a multi-front battle to prevent authorized access to networks and the information contained within. Network administrators and security personnel face external and internal efforts to profile their network in order to gain exploitable information about the network. We described the general categories by which most threat agents may be grouped. We explained how these agents utilize malicious software and malicious activities to advance toward their goal of gaining and maintaining access to your network.

In Chapter III, we detailed our design considerations. We presented a perceived misunderstanding of the defensive readiness of individual DoD networks connected to the GIG. In the pursuit for increased offensive and defensive capabilities to produce strategic affects in cyberspace, the lack of ability for unit network administrators to provide for the self-defense of their operational networks gets little attention. We explained how MAST provides unit commanders the ability to train their network administrators to recognize and respond to malicious behavior that penetrates their network through the imperfections in its armor. To do this, MAST must deliver simulations of real malicious activity in a real computing environment in a safe and scalable manner. We provided detailed information on how these factors were accounted for in the selection and design of the COMPOSE testing environments.

In Chapter IV, we discussed the methodology and results of our experimentation. We described the research conducted before testing began to ensure our test environments provided the shipboard network replication our hypothesis rested on. Most of this description addressed the creation and configuration of the MAST Test Bed, since the NCOR testing environment is a certified and accredited COMPOSE network. During Simware module pre-test development, we worked with Belli to verify that the simulation delivered during execution matched the intent of the module. Once armed with proven Simware modules, we conducted our computability experiments on the MAST Test Bed.

B. CONCLUSIONS

Our successful testing of MAST's operational features on the MAST Test Bed verified MAST's ability to produce realistic, tailored, scalable simulation of malicious activity without degrading any of COMPOSE capabilities. This ability was validated by the successful outcome of our NCOR experiments. The most important control for our experiments was the presence of a proper installation and configuration of the COMPOSE environment. We were fortunate to receive a more recent version than that used in the virtual tests of MAST (version 3.5 versus version 3.0.1). One pre-experimental variable in our experiment was installation and configuration errors due to inexperience with the process. A shipboard instance of the COMPOSE version 3.5 environment was installed by a SPAWAR COMPOSE Installation Engineer who had experience with several installations experience. Although lacking in the type of experience that may minimize errors, three factors reduced the window for error this pre-experimental variable introduced. For both COMPOSE and CND-OSE, the installation and configuration process was guided by SPAWAR manuals, aided by our combined operational experience with both operating system environments, and augmented by distance support from COMPOSE and CND-OSE Installation Engineers.

A key lesson learned from our research is the importance of the SPAWAR Command, Control, Communications, Computers, and Intelligence (C4I) Chair at NPS. All technical assistance provided by PMW-130 and PMW-160 would normally require a contract between the PMW offices and the recipient of the technical support hours.

Instead, Dr. Rachelle Goshorn, NPS C4I Chair, worked with the leadership at SPAWAR to identify us as a research effort. The costs associated with the technical support we received were funded by SPAWAR and significantly minimized thanks to her intervention and guidance.

Another key lesson learned is the benefit gained by interacting with not only the NCOR team but also NIOC Norfolk as a whole during our NCOR testing phase. Having the opportunity to display MAST in operation, explain its purpose, and solicit feedback from one the U.S. Navy commands responsible for training sailors to support the DCO mission and assess the operational readiness of Navy networks was invaluable. We present our case for continued collaboration with the NCOR in the next section.

The success of this research answered our thesis. MAST can function on a LAN using the COMPOSE operating system with high confidence that it will respond as directed. After successfully reaching the same conclusion at NCOR, our results from the MAST Test Bed experiments were reinforced.

C. FUTURE WORK

1. Continued Collaboration with the NCOR

The NCOR is certified and accredited to connect to the GIG. As such, NIOC Norfolk utilizes the NCOR to conduct re-familiarization training of its previous students at their current commands remotely via Virtual Private Networks (VPN) [38]. This ability to establish secure connections with geographically-dispersed networks presents an excellent opportunity to conduct future MAST experimentation. A VPN between MAST and the NCOR would enable the connection of MAST to multiple networks. This remote connection would allow for future functionality tests of the SG server and its capabilities. A VPN with the NCOR would enable testing of the SG server's ability to load, define, execute, monitor, and secure a simulation scenario remotely on two networks (CVN-65 domain and CG-48 domain) simultaneously. Additionally, it would provide the ability to test and develop Simware modules that deliver simulation of externally generated malicious activity. Collaboration with the NCOR via VPN on future MAST research

would bring greater realism to the experimentation while significantly reducing the amount of associated travel funds.

2. Further Advancements in Simulation

Chapter II presented threats faced by any network connected to cyberspace. No Simware module library would ever be able to produce scenarios necessary to simulate all threats; not all threats are known. Also, as explained in Chapter III, when training, it is not reasonable to simulate malicious activity beyond the ability of the network administrator or his relatively basic security tools to detect. However, we propose several additional avenues of simulation to improve upon MAST's relevance to today's larger, more well-known threats.

We propose the creation of one or more malicious websites. A malicious website would improve upon MAST's ability to simulate social engineering attacks. In keeping with the scalability criteria of MAST, these malicious websites would act as blank canvas upon which the scenario developer may select the malicious threat to present the training audience. We explained in Chapter II how a malicious e-mail may be sent to the training audience with a hyper-link, which would direct the user to a malicious website containing drive-by downloads. Figure 14 is a revision of the training scenario presented in Chapter III (Figure 5). The highlighted sections show how a malicious website would support the delivery of simulated malicious behavior for training and assessment purposes.

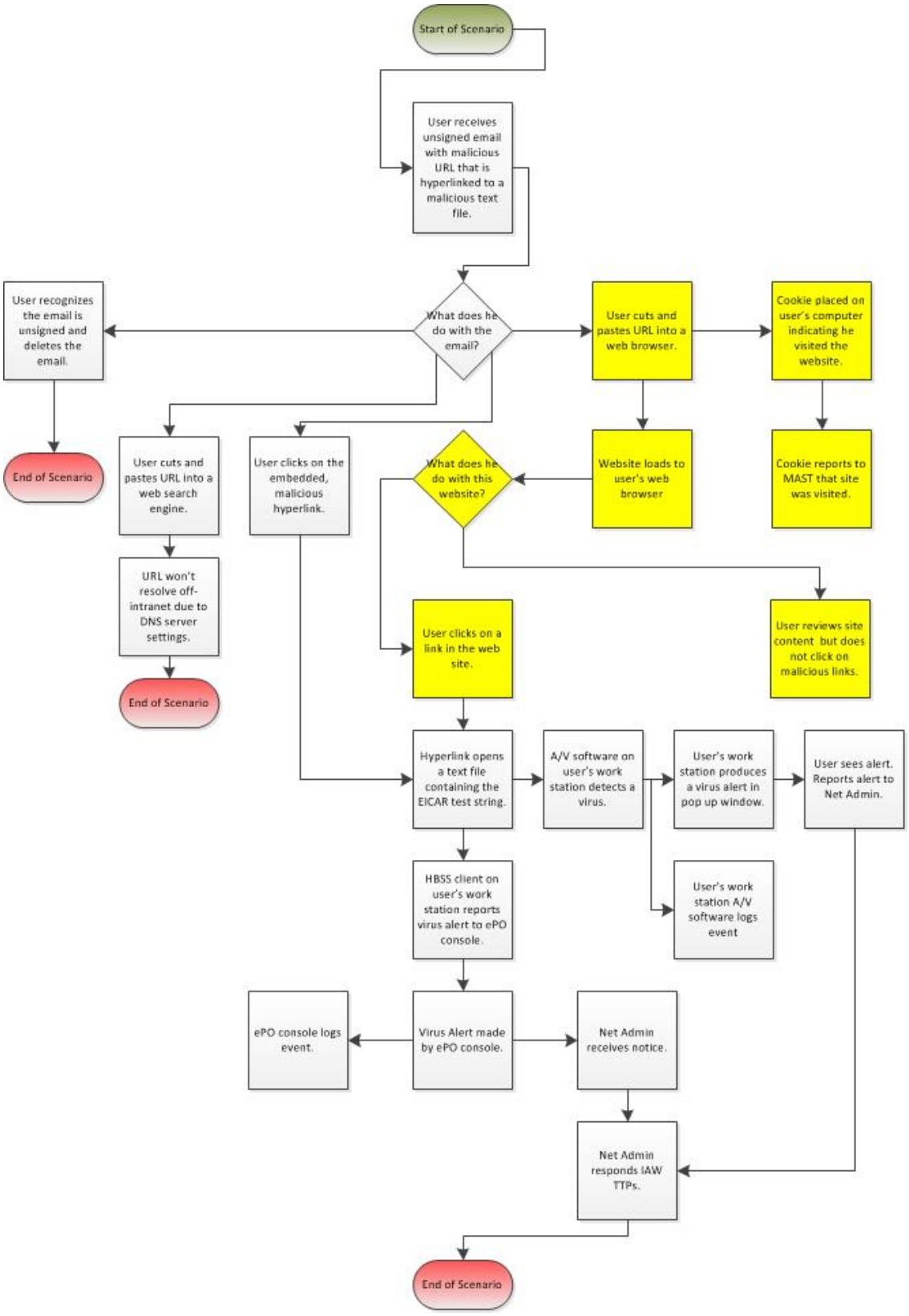


Figure 15. MAST Training Scenario Employing Malicious website

As a safety feature, this website would be provided with an internal IP address that would not resolve outside of the participating operational network. This provides containment should the e-mail be forwarded to an external e-mail account. The website would either be hosted by the web server of the participating network or networks. To avoid configuration errors and to afford MAST more control over its simulations, developing a web server as part of the deployed MAST package may be in order. By utilizing a MAST web server, the command employing MAST for remote purposes could define the web server IP address assignment and any necessary host network configuration settings in the Memorandum of Agreement (MOA) signed by the participating command or commands prior to commencement of MAST simulation. This alleviates the participating command of any responsibility for the web server's settings. MAST, to include the MAST web server, would be deployed as a package configured and ready for use.

A recommendation received during our testing at the NCOR was to implement a simulation of a botnet Command and Control (C2) server [32]. Bots, also discussed in Chapter II, require communication with their master in order to receive and execute their master's bidding. In order to hide their identity, botnet masters will utilize C2 servers separate from them to automatically provide command and control to their bots. The bots are designed to deploy on their host workstation and, utilizing the host's access to the Internet, call out to the pre-programmed IP address of the C2 server. This callout signals their existence and queries for instructions. Regarding simulation of a C2 server communicating with a bot, there are several options. One is to configure the MAST web server to send traffic to legitimate IP addresses from a spoofed, external IP address. Another would be to create a Simware module that requires the host workstation to send packets to an IP address that will not resolve off the host network. For example, the IP address is a private IP address not associated with the host network but blocked by the host network's outbound firewall settings. In both cases, the intent is not to pursue the ability to send externally-generated simulation of botnet C2 traffic, which might pose a

security threat to the host networks. Instead, the intent is to create internally-generated traffic that would be detected and identified as potentially malicious, either by log analysis or by HIPS alert.

3. Fleet Operational Tests of MAST

To date, all MAST research, both ours and that summarized in Chapter II, shaped and guided the functionality of MAST toward its end objective of supporting the war fighter on their operational network by providing simulated malicious behavior for the purpose of enhancing their command's operational readiness and security posture in cyber space. The next logical step in progressing MAST toward operational deployment in the Fleet is to test it on an operational network with human interaction and participation. TRIDENT WARRIOR, the DON vehicle for conducting at-sea experimentation of both technology and the tactics used in employing them [39], is an excellent candidate for this next step. Testing MAST on an actual shipboard network would further validate the results of our research as well as provide the opportunity to collect end-user feedback.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] W. E. Leigher. (2011, Feb.). *Learning to operate in cyberspace* [Online]. Available: http://www.usni.org/magazines/proceedings/2011–02/learning-operate-cyberspace

[2] W. R. Taff Jr., and P. M. Salevski, "Malware mimics for network security assessment," M.S. thesis, Dept. Comput. Sci., Naval Postgraduate School, Monterey, CA, 2011.

[3] J. M. Neff, "Verification and validation of the Malicious Activity Simulation Tool (MAST) for network administrator training and evaluation," M.S. thesis, Dept. Comput. Sci., Naval Postgraduate School, Monterey, CA, 2012.

[4] R. Longoria, "Scalability assessments for the Malicious Activity Simulation Tool (MAST) for network administrator training and evaluation," M.S. thesis, Dept. Comput. Sci., Naval Postgraduate School, Monterey, CA, 2012.

[5] J. Hammond, "Malicious behavior expansion," presented at Program Sponsor Brief, Naval Postgraduate School, Monterey, CA, 2012.

[6] N. Hayes, "A definitive interoperability test methodology for the Malicious Activity Simulation Tool (MAST)," M.S. thesis, Dept. Comput. Sci., Naval Postgraduate School, Monterey, CA, 2013.

[7] G. Belli, "Extensible Simware architecture for flexible training scenarios," work in progress, Dept. Comput. Sci., Naval Postgraduate School, Monterey, CA.

[8] E. Lowney, "Architecting a communications protocol for multiple MAST scenarios," work in progress, Dept. Comput. Sci., Naval Postgraduate School, Monterey, CA.

[9] Department of the Navy, (2013, April) *Operational test director's manual* [Online]. Available: http://www.public.navy.mil/cotf/Pages/OTDManual.aspx

[10] Federal Information Security Management Act of 2002, Public Law 107–347 § 3542.

[11] S. Harris, *CISSP All-in-One Exam Guide*, Fifth Edition. New York: McGraw Hill, 2010.

[12] K. Burton. (n.d.). *The Conficker Worm* [Online]. Available: https://www.sans.org/security-resources/malwarefaq/conficker-worm.php

[13] M. Hypponen. (2013, March 30). *Where we are and where we are going* [Online]. Available:http://www.hackinparis.com/slides/hip2k12/Mikko%20H-Keynote.pdf, Jun. 21, 2012

[14] Internet Crime Compliant Center, "2011 Internet crime report," National White Collar Crime Center, Glen Allen, VA. May 2012.

[15] F. Paget, "Hacktivism," McAfee, Inc., Santa Clara, CA, Nov. 2008. [Online]. Available: http://www.mcafee.com/us/resources/white-papers/wp-hacktivism.pdf

[16] Mandiant. (2013, February 18). *APT1: Exposing one of China's cyber espionage units.* [Online]. Available: https://intelreport.mandiant.com/Mandiant_APT1_Report.pdf

[17] T. Nash, "An undirected attack against critical infrastructure: a case study for improving your control system security." U.S. Department of Homeland Security Vulnerability & Risk Assessment Program (VRAP), Livermore, CA, Sep. 2005.

[18] M. Dalla Preda. (2006, August 22). *Hunting obfuscated malware by abstract interpretation* [Online] Available: http://profs.sci.univr.it/~dallapre/hunting.pdf

[19] M. Sikorski and A. Honig, "Malware analysis primer," in *Practical Malware Analysis*, San Francisco, CA: No Starch Press, 2012, pp. 3–4.

[20] T. Eisenberg *et al.*, "The Cornell commission: On Morris and the worm," *Communications of the ACM*, vol. 32, no. 6, pp. 706–707, Jun. 1989.

[21] J. Kong, "Introduction," in *Designing BSD Rootkits: An Introduction to Kernel Hacking*, San Francisco, CA: No Starch Press, 2007, pp. xvi.

[22] *Certified Ethical Hacker: Ethical Hacking and Countermeasures*, Courseware Guide v7.1 Volume 1, EC-Council USA, Albuquerque, NM, 2011, pp. 16–19.

[23] Information Sciences Institute, Univ. Southern California. (1981, Sep.). *Transmission Control Protocol: DARPA Internet Program Protocol Specification,IETF RFC 793* [Online] Available: https://www.ietf.org/rfc/rfc793.txt.

[24] W3Schools. (n.d.). *HTML <iframe> Tag – www.w3schools.com* [Online]. Available: http://www.w3schools.com/tags/tag_iframe.asp.

[25] K. Presti (2012, Nov. 28). *Malicious pop-ups promise browser updates but spread malware* [Online]. Avialabile: http://www.crn.com/news/security/240142710/malicious-pop-ups-promise-browser-updates-but-spread-malware.htm

[26] U.S. Department of Defense, “2010 Defense Quadrennial Review,” Washington, D.C.: Government Printing Press, Feb. 2010.

[27] U.S. Government Accountability Office. (2011, Jun. 20). *Defense department cyber efforts: More detailed guidance needed to ensure military services develop appropriate cyberspace capabilities* [Online]. Available: http://www.gao.gov/assets/320/318609.html

[28] J. A. Powell, Navy Cyberspace Operations Range (NCOR) overview. [Personal e-mail] (2013, Jan. 17).

[29] *Communications Task Order (CTO) 07–15*, Joint Task Force-Global Network Operations, Apr. 2008.

[30] European Expert Group for IT-Security. (n.d.). *Anti-malware test file - intended use* [Online]. Available: http://www.eicar.org/86–0-Intended-use.html

[31] S. Hood, private communication, Oct. 2012.

[32] S. Calhoun and J. Powell, private communication, Aug. 2012.

[33] Impulse Point. (n.d.). *SafeConnect network access control* [Online]. Available: http://www.impulse.com/safeconnect/

[34] C. T. Wai. (2002). *Conducting a penetration test on an organization* [Online]. Available: https://www.sans.org/reading-room/whitepapers/auditing/conducting-penetration-test-organization-67

[35] G. Belli, private communication, Apr. 2013.

[36] A. Bennieston. (2009). *NMAP – a stealthy port scanner* [Online]. Available: http://NMAP.org/bennieston-tutorial/

[37] J. A. Powell, private communication, May 2013.

[38] S. Calhoun. private communication, Jan. 2013.

[39] E. Doll. (2011). *Trident Warrior 2011* [Online]. Available: http://www.DONcio.navy.mil/CHIPS/ArticleDetails.aspx?id=3025

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California